



TUGAS AKHIR - KI141502

Rancang Bangun Aplikasi Sistem Basis Data *Online Judge* (SBDOJ) untuk Proses Pembelajaran Mata Kuliah Sistem Basis Data di Departemen Teknik Informatika ITS

DEMSY IMAN MUSTASYAR
NRP 5113 100 015

Dosen Pembimbing 1
RIZKY JANUAR AKBAR, S.Kom., M.Eng.

Dosen Pembimbing 2
NURUL FAJRIN A., S.Kom, M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017



TUGAS AKHIR - KI141502

**Rancang Bangun Aplikasi Sistem Basis Data
Online judge (SBDOJ) untuk Proses
Pembelajaran Mata Kuliah Sistem Basis
Data di Departemen Teknik Informatika ITS**

DEMSY IMAN MUSTASYAR
NRP 5113 100 015

Dosen Pembimbing 1
RIZKY JANUAR AKBAR, S.Kom., M.Eng.

Dosen Pembimbing 2
NURUL FAJRIN A., S.Kom, M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]



UNDERGRADUATE THESIS - KI141502

Application Design of Database System Online Judge (SBDOJ) for Learning Process of Database System Course at The Department of Informatics Engineering ITS

DEMSY IMAN MUSTASYAR
NRP 5113 100 015

Supervisor 1
RIZKY JANUAR AKBAR, S.Kom., M.Eng.

Supervisor 2
NURUL FAJRIN A., S.Kom, M.Sc.

INFORMATICS ENGINEERING DEPARTMENT
Information Technology Faculty
Sepuluh Nopember Institute of Technology
Surabaya 2017

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

RANCANG BANGUN APLIKASI SISTEM BASIS DATA ONLINE JUDGE (SBDOJ) UNTUK PROSES PEMBELAJARAN MATA KULIAH SISTEM BASIS DATA DI DEPARTEMEN TEKNIK INFORMATIKA ITS

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Algoritma dan Pemrograman
Program Studi S-1 Departemen Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :
DEMSY IMAN MUSTASYAR
NRP. 5113 100 015

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Rizky Januar Akbar, S.Kom, M.Eng.
NIP. 198701032014041001 (Pembimbing 1)
2. Nurul Fajrin A., S.Kom, M.Sc.
NIP. 198607222015042003 (Pembimbing 2)

SURABAYA
JUNI 2017

[Halaman ini sengaja dikosongkan]

**RANCANG BANGUN APLIKASI SISTEM BASIS DATA
ONLINE JUDGE (SBDOJ) UNTUK PROSES
PEBELAJARAN MATA KULIAH SISTEM BASIS DATA
DI DEPARTEMEN TEKNIK INFORMATIKA ITS**

Nama : Demy Iman Mustasyar
NRP : 5113100015
Jurusan : Teknik Informatika
Fakultas Teknologi Informasi ITS
Dosen Pembimbing 1 : Rizky Januar Akbar, S.Kom., M.Eng.
Dosen Pembimbing 2 : Nurul Fajrin A., S.Kom., M.Sc.

ABSTRAK

Saat ini, praktikum Sistem Basis Data di Departemen Teknik Informatika ITS Surabaya masih dilaksanakan secara manual. Prosesnya adalah ketika praktikan selesai mengerjakan sebuah soal, praktikan diharuskan memanggil asisten untuk dicek dan dinilai. Hal tersebut tentu memakan waktu banyak dan juga memberatkan asisten. Oleh karena itu, diperlukan sebuah aplikasi yang membantu praktikum Sistem Basis Data untuk mengecek dan memberi nilai kepada jawaban para praktikan.

Aplikasi Online Judge dalam tugas akhir ini memanfaatkan Laravel sebagai kerangka kerja pemrograman. Penggunaan Laravel diharapkan dapat mempercepat proses pengembangan dan mempermudah proses perawatan sistem. Untuk proses penilaiannya, diperlukan sebuah proses yang bisa berjalan di balik layar, atau yang dinamakan Grader. Grader tersebut menggunakan bahasa Python agar mampu berjalan di balik layar.

Pengujian difokuskan kepada proses penilaian atau judgement. Pengujian dilakukan dengan melakukan uji coba beberapa jenis query. Jenis-jenis query tersebut didapatkan dari praktikum Sistem Basis Data dengan menggunakan aplikasi Online Judge ini. Praktikan dan asisten akan diminta untuk melakukan praktikum dengan menggunakan aplikasi ini sehingga pengujian dapat dilakukan dengan situasi yang nyata. Pengujian dilakukan untuk mengetahui keberhasilan grader dalam menangani proses penilaian query jawaban praktikan.

Kata Kunci: Sistem Basis Data, Online Judge, Grader, Kerangka Kerja Laravel, Python.

**APPLICATION DESIGN OF DATABASE SYSTEM
ONLINE JUDGE (SBDOJ) FOR LEARNING PROCESS
OF DATABASE SYSTEM COURSE AT THE
DEPARTMENT OF INFORMATICS ENGINEERING
ITS**

Name : Demsy Iman Mustasyar
NRP : 5113100015
Department : Department of Informatics
Faculty of Information Technology ITS
Supervisor I : Rizky Januar Akbar, S.Kom., M.Eng.
Supervisor II : Nurul Fajrin A., S.Kom., M.Sc.

ABSTRACT

Nowadays, Informatics Department in ITS Surabaya still implement the practice of Database System course manually. The students must answer the questions, then the assistants must check and give a mark to their query. This process takes a long time and also complicates the assistants. Therefore, they need an application that can help the practice of Database System course for checking and judging the query of the students.

The Online Judge Application in this undergraduate thesis utilize Laravel as programming frameworks. The use of Laravel Framework is expected to accelerate the development process and simplify the process of system maintenance. For the judgment process, it needs a process that can run in background, or so-called Grader. This Grader use Python programming language in order to run in background.

The tests of this application focus on the judgement process. The testing was performed by testing several types of queries. That queries were obtained from the practice of Database System course by using this Online Judge Application.

The students and assistants were asked to do the practice by using this application so the testing can be done with real situation. The test was conducted to determine the success of Grader in handling the judgement process of student's queries.

Keywords: Database System, Online Judge, Grader, Laravel Framework, Python.

KATA PENGANTAR

Alhamdulillahirabbil'alamin. Puji syukur kehadiran Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul **“RANCANG BANGUN APLIKASI SISTEM BASIS DATA *ONLINE JUDGE* (SBDOJ) UNTUK PROSES PEMBELAJARAN MATA KULIAH SISTEM BASIS DATA DI DEPARTEMEN TEKNIK INFORMATIKA ITS”**.

Pengerjaan tugas akhir ini merupakan suatu kesempatan yang sangat baik bagi penulis, karena dengan ini penulis dapat belajar lebih banyak untuk memperdalam dan meningkatkan apa yang telah didapatkan selama menempuh perkuliahan di Teknik Informatika ITS.

Selesainya tugas akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak, sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada:

1. Abah, Mama dan Mbak Priesty yang selalu mendukung dan mendoakan yang terbaik.
2. Bapak Rizky Januar Akbar, S.Kom., M.Eng. selaku pembimbing I yang selalu membimbing dan memotivasi penulis selama pengerjaan tugas akhir.
3. Ibu Nurul Fajrin A., S.Kom., M.Sc. selaku pembimbing II yang juga telah membantu, membimbing dan memberikan semangat selama pengerjaan tugas akhir.
4. Pak Darlis selaku Kepala Departemen Teknik Informatika ITS, Pak Onggo selaku koordinator TA dan Bu Sarwosri selaku dosen wali penulis dan segenap dosen Teknik Informatika yang telah memberikan ilmunya.
5. Galuh, selaku orang yang selalu memberikan inspirasi, semangat, motivasi dan bantuan selama pengerjaan tugas akhir ini.
6. Mbak Rina, selaku orang yang tidak henti-hentinya memberi motivasi, bantuan, dan ide selama pengerjaan tugas akhir ini.

7. Bilfash, Luffi dan Yohana, sebagai tim Kadut yang telah menemani, mendukung dan menyemangati penulis selama 4 tahun kuliah di Teknik Informatika ITS.
8. Rekan, teman dan sahabat administrator Laboratorium Algoritma & Pemrograman; Mas Udin, Mas Hanif, Mbak Shoffi, Mba Rina, Yohana, , Luffi, Ridho, Arvi, Rani, Sabila, Aldi, Dimas, Mumul, Rozana, Atul, Reinardus, Abyan dan mas Khairy, yang telah memberikan “semangat” dan selalu menemani serta mewarnai hari-hari penulis di laboratorium.
9. Mas Yunus, Mas Indra, Mas Mahen dan Mas Bustan selaku administrator Laboratorium Algoritma & Pemrograman periode 2014/2015 yang dengan sabar mengajari dan membagikan ilmunya kepada penulis.
10. Ghulam, Adi, Fajar, Cayza, Apip, Riska, Rei, Ihsan, Razi, Anip, selaku teman bermain DotA 2 yang membuang banyak waktu selama berkuliah di Teknik Informatika ITS.
11. Ine, Risma, Burhan, Sari, Putri, Gilang, John, Cao, Fablius, Ronald, Luthfie, Yahya, Udin, Kepek, Lusi, Renanda, Andi, Nindy, Anwar, Uul, Fakhri, Nela, Tities, dan Inyas yang selalu memberikan semangat, saran, motivasi dan kenangan-kenangan indah selama berkuliah di Teknik Informatika ITS.
12. Teman-teman angkatan 2013 yang telah membantu, berbagi ilmu, dan memberi motivasi kepada penulis selama masa perkuliahan.
13. Serta semua pihak yang turut membantu penulis dalam menyelesaikan tugas akhir ini.

Penulis menyadari bahwa tugas akhir ini masih memiliki banyak kekurangan, sehingga dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depannya.

Surabaya, Juni 2017

Penulis

DAFTAR ISI

LEMBAR PENGESAHAN.....	v
ABSTRAK	vii
ABSTRACT	ix
KATA PENGANTAR.....	xi
DAFTAR ISI.....	xiii
DAFTAR GAMBAR	xvii
DAFTAR TABEL	xxi
DAFTAR KODE SUMBER	xxiii
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Permasalahan	2
1.3 Batasan Masalah	2
1.4 Tujuan	3
1.5 Manfaat	3
1.6 Metodologi	3
1.7 Sistematika Penulisan	4
BAB II TINJAUAN PUSTAKA	7
2.1 <i>Online Judge</i>	7
2.2 <i>Database</i>	8
2.3 SQL	8
2.3.1 <i>Simple Query</i>	8
2.3.2 <i>Medium Query</i>	9
2.3.3 <i>Complex Query</i>	10
2.4 Kerangka Kerja Laravel	11
2.5 PHP	12
2.6 OracleSQL	13
BAB III ANALISIS DAN PERANCANGAN SISTEM.....	15
3.1 Analisis Sistem.....	15
3.1.1 Deskripsi Umum Aplikasi	15
3.1.2 Analisis Proses Bisnis	16
3.1.3 Analisis Kebutuhan Sistem	18
3.1.4 Analisis Aktor	20
3.2 Perancangan Sistem	20

3.2.1 Kasus Penggunaan.....	20
3.2.1.1 Kasus Penggunaan Mengelola Data Praktikan.....	21
3.2.1.2 Kasus Penggunaan Mengelola Data <i>Database</i>	26
3.2.1.3 Kasus Penggunaan Mengelola Jenis <i>Database</i>	30
3.2.1.4 Kasus Penggunaan Mengelola <i>Grader</i>	32
3.2.1.5 Kasus Penggunaan Mengelola <i>Event</i>	35
3.2.1.6 Kasus Penggunaan Melihat <i>Submissions</i>	52
3.2.1.7 Kasus Penggunaan Melihat <i>Scoreboard</i>	55
3.2.1.8 Kasus Penggunaan Menjawab Soal <i>Event</i>	56
3.2.1.9 Kasus Penggunaan Mengelola Akun	57
3.2.1.10 Kasus Penggunaan Me-reset <i>Password</i>	60
3.2.2 Perancangan Diagram Kelas.....	61
3.2.2.1 Kelas Role	65
3.2.2.2 Kelas Users	65
3.2.2.3 Kelas DBVersionParameter	66
3.2.2.4 Kelas DBVersion	66
3.2.2.5 Kelas ListDBParameter	66
3.2.2.6 Kelas ListDB	67
3.2.2.7 Kelas Event	67
3.2.2.8 Kelas EventTutorial.....	68
3.2.2.9 Kelas Question.....	68
3.2.2.10 Kelas Tutorial.....	69
3.2.2.11 Kelas Category.....	69
3.2.2.12 Kelas Submission	70
3.2.2.13 Kelas SubmissionTutorial	70
3.2.2.14 Kelas AccountController.....	71
3.2.2.15 Kelas DBVersionController	71
3.2.2.16 Kelas <i>DatabaseController</i>	71
3.2.2.17 Kelas EventController.....	72
3.2.2.18 Kelas QuestionController	72
3.2.2.19 Kelas EventTutorialController	73
3.2.2.20 Kelas CategoryController.....	73
3.2.2.21 Kelas TutorialController	74

3.2.2.22 Kelas AdminController	74
3.2.3 Perancangan Arsitektur Aplikasi	74
BAB IV IMPLEMENTASI	77
4.1 Lingkungan Implementasi	77
4.1.1 Lingkungan Implementasi Perangkat Keras	77
4.1.2 Lingkungan Implementasi Perangkat Lunak	77
4.2 Implementasi Perangkat Lunak	78
4.2.1 Implementasi Proses Mengelola Data Praktikan	78
4.2.2 Implementasi Proses Mengelola Data <i>Database</i>	78
4.2.3 Implementasi Proses Mengelola Jenis <i>Database</i>	78
4.2.4 Implementasi Proses Mengelola <i>Grader</i>	79
4.2.5 Implementasi Proses Mengelola <i>Event</i>	81
4.2.6 Implementasi Proses Mengelola Soal	81
4.2.7 Implementasi Proses Melihat <i>Submissions</i>	82
4.2.8 Implementasi Proses Melihat <i>Scoreboard</i>	83
4.2.9 Implementasi Proses Menjawab Soal <i>Event</i>	86
4.2.10 Implementasi Proses Mengelola Akun	87
4.2.11 Implementasi Proses Me-reset <i>Password</i>	88
4.3 Implementasi Antarmuka Pengguna	91
4.3.1 Antarmuka Mengelola Data Praktikan	91
4.3.2 Antarmuka Mengelola <i>Database</i>	94
4.3.3 Antarmuka Mengelola Jenis <i>Database</i>	96
4.3.4 Antarmuka Mengelola <i>Grader</i>	98
4.3.5 Antarmuka Mengelola <i>Event</i>	98
4.3.6 Antarmuka Mengelola Soal	101
4.3.7 Antarmuka Melihat <i>Submissions</i>	103
4.3.8 Rancangan Antarmuka Melihat <i>Scoreboard</i>	104
4.3.9 Antarmuka Menjawab Soal <i>Event</i>	104
4.3.10 Antarmuka Mengelola Akun	105
4.3.11 Antarmuka Me-reset <i>Password</i>	107
BAB V PENGUJIAN DAN EVALUASI	109
5.1 Lingkungan Pelaksanaan Pengujian	109
5.2 Pengujian <i>User Acceptance Test</i>	109
5.3 Pengujian Sistem	111
5.3.1 Pengujian Tingkat Keberhasilan <i>Grader</i>	111

5.3.1.1	Pengujian <i>Grader</i> Versi I.....	112
5.3.1.2	Pengujian <i>Grader</i> Versi II.....	113
5.4	Evaluasi Pengujian	119
5.4.1	Evaluasi Pengujian <i>User Acceptance Test</i>	119
5.4.2	Evaluasi Pengujian Tingkat Keberhasilan	120
	BAB VI KESIMPULAN DAN SARAN	121
6.1	Kesimpulan	121
6.2	Saran.....	122
	DAFTAR PUSTAKA.....	123
	LAMPIRAN A. KODE SUMBER.....	125
	LAMPIRAN B. KUESIONER UAT	141
	BIODATA PENULIS.....	148

DAFTAR GAMBAR

Gambar 2.1 Hasil Simple Query	9
Gambar 2.2 Hasil Medium Query	10
Gambar 2.3 Hasil Complex Query	11
Gambar 3.1 Deskripsi Umum Perangkat Lunak.....	16
Gambar 3.2 Proses Bisnis Sistem Basis Data Online judge pada Asisten.....	17
Gambar 3.3 Proses Bisnis Sistem Basis Data Online judge pada Praktikan.....	18
Gambar 3.4 Diagram Kasus Penggunaan	20
Gambar 3.5 Diagram Aktivitas Menambah Data Praktikan.....	22
Gambar 3.6 Diagram Aktivitas Mengubah Data Praktikan.....	23
Gambar 3.7 Diagram Aktivitas Meghapus Data Praktikan	24
Gambar 3.8 Diagram Aktivitas Menambah Data Praktikan dengan File Excel.....	26
Gambar 3.9 Diagram Aktivitas Menambah Data Database	27
Gambar 3.10 Diagram Aktivitas Mengubah Data Database	28
Gambar 3.11 Diagram Aktivitas Menghapus Data Database.....	29
Gambar 3.12 Diagram Aktivitas Menambah Jenis Database.....	31
Gambar 3.13 Diagram Aktivitas Mengubah Jenis Database.....	32
Gambar 3.14 Diagram Aktivitas Memulai Grader	33
Gambar 3.15 Diagram Aktivitas Menghentikan Grader	34
Gambar 3.16 Diagram Aktivitas Menambah Praktikum.....	35
Gambar 3.17 Diagram Aktivitas Mengubah Praktikum.....	36
Gambar 3.18 Diagram Aktivitas Menghapus Praktikum	37
Gambar 3.19 Diagram Aktivitas Menambah Tutorial.....	38
Gambar 3.20 Diagram Aktivitas Mengubah Tutorial.....	39
Gambar 3.21 Diagram Aktivitas Menghapus Tutorial	40
Gambar 3.22 Diagram Aktivitas Menambah Kategori Tutorial..	41
Gambar 3.23 Diagram Aktivitas Mengubah Kategori Tutorial...	43
Gambar 3.24 Diagram Aktivitas Menghapus Kategori Tutorial.	44
Gambar 3.25 Diagram Aktivitas Menambah Soal Praktikum.....	45
Gambar 3.26 Diagram Aktivitas Mengubah Soal Praktikum.....	47
Gambar 3.27 Diagram Aktivitas Menghapus Soal Praktikum	48

Gambar 3.28 Diagram Aktivitas Menambah Soal Tutorial	49
Gambar 3.29 Diagram Aktivitas Mengubah Soal Tutorial.....	50
Gambar 3.30 Diagram Aktivitas Menghapus Soal Tutorial	52
Gambar 3.31 Diagram Aktivitas Melihat Submissions untuk Asisten	53
Gambar 3.32 Diagram Aktivitas Melihat Submissions untuk Praktikan.....	54
Gambar 3.33 Diagram Aktivitas Melihat Scoreboard	55
Gambar 3.34 Diagram Aktivitas Menjawab Soal Event	56
Gambar 3.35 Diagram Aktivitas Mengubah Data Diri Praktikan.....	57
Gambar 3.36 Diagram Aktivitas Mengubah Password	59
Gambar 3.37 Diagram Aktivitas Me-reset Password	60
Gambar 3.38 Diagram Kelas SBDOJ	63
Gambar 3.39 Kelas Role.....	65
Gambar 3.40 Kelas Users	65
Gambar 3.41 Kelas DBVersionParameter	66
Gambar 3.42 Kelas DBVersion.....	66
Gambar 3.43 Kelas ListDParamter.....	66
Gambar 3.44 Kelas ListDB	67
Gambar 3.45 Kelas Event.....	67
Gambar 3.46 Kelas EventTutorial	68
Gambar 3.47 Kelas Question.....	68
Gambar 3.48 Kelas Tutorial	69
Gambar 3.49 Kelas Category.....	69
Gambar 3.50 Kelas Submission.....	70
Gambar 3.51 Kelas SubmissionTutorial.....	70
Gambar 3.52 Kelas AccountController	71
Gambar 3.53 Kelas DBVersionController.....	71
Gambar 3.54 Kelas DatabaseController	71
Gambar 3.55 Kelas EventController	72
Gambar 3.56 Kelas QuestionController	72
Gambar 3.57 Kelas EventTutorialController.....	73
Gambar 3.58 Kelas CategoryController	73
Gambar 3.59 Kelas TutorialController.....	74
Gambar 3.60 Kelas AdminController.....	74

Gambar 3.61 Arsitektur Aplikasi	75
Gambar 4.1 Antarmuka Mengelola Data Praktikan	91
Gambar 4.2 Antarmuka Menambah Data Praktikan	92
Gambar 4.3 Antarmuka Mengubah Data Praktikan	92
Gambar 4.4 Antarmuka Menghapus Data Praktikan.....	93
Gambar 4.5 Antarmuka Upload File Excel	93
Gambar 4.6 Antarmuka Mengelola Database	94
Gambar 4.7 Antarmuka Menambah Database.....	95
Gambar 4.8 Antarmuka Mengubah Database	95
Gambar 4.9 Antarmuka Menghapus Database.....	96
Gambar 4.10 Antarmuka Mengelola Jenis Database	96
Gambar 4.11 Antarmuka Menambah Jenis Database.....	97
Gambar 4.12 Antarmuka Mengubah Jenis Database	97
Gambar 4.13 Antarmuka Mengelola Grader	98
Gambar 4.14 Antarmuka Mengelola Event.....	99
Gambar 4.15 Antarmuka Menambah Event.....	99
Gambar 4.16 Antarmuka Mengubah Event.....	100
Gambar 4.17 Antarmuka Menghapus Event	100
Gambar 4.18 Antarmuka Mengelola Kategori Tutorial	101
Gambar 4.19 Antarmuka Menambah Kategori Tutorial	101
Gambar 4.20 Antarmuka Mengelola Soal	102
Gambar 4.21 Antarmuka Tambah Soal.....	102
Gambar 4.22 Antarmuka Ubah Soal	103
Gambar 4.23 Antarmuka Hapus Soal.....	103
Gambar 4.24 Antarmuka Daftar Submissions.....	104
Gambar 4.25 Antarmuka Scoreboard	105
Gambar 4.26 Antarmuka Menjawab Soal Event	105
Gambar 4.27 Antarmuka Mengubah Data Diri Praktikan	106
Gambar 4.28 Antarmuka Mengubah Password.....	106
Gambar 4.29 Antarmuka Memasukkan NRP.....	107
Gambar 4.30 Antarmuka Reset Password.....	107

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

Tabel 3.1 Daftar Kebutuhan Fungsional Sistem.....	19
Tabel 3.2 Keterangan Kode Kasus Penggunaan.....	21
Tabel 3.3 Spesifikasi Kasus Penggunaan Menambah Data Praktikan.....	22
Tabel 3.4 Spesifikasi Kasus Penggunaan Mengubah Data Praktikan	23
Tabel 3.5 Spesifikasi Kasus Penggunaan Menghapus Data Praktikan.....	24
Tabel 3.6 Spesifikasi Kasus Penggunaan Menambah Data Praktikan dengan File Excel.....	25
Tabel 3.7 Spesifikasi Kasus Penggunaan Menambah Data Database	27
Tabel 3.8 Spesifikasi Kasus Penggunaan Mengubah Data Database	28
Tabel 3.9 Spesifikasi Kasus Penggunaan Menghapus Data Database	29
Tabel 3.10 Spesifikasi Kasus Penggunaan Menambah Jenis Database	30
Tabel 3.11 Spesifikasi Kasus Penggunaan Mengubah Data Jenis Database	31
Tabel 3.12 Spesifikasi Kasus Penggunaan Memulai Grader.....	33
Tabel 3.13 Spesifikasi Kasus Penggunaan Mengubah Data Praktikan.....	34
Tabel 3.14 Spesifikasi Kasus Penggunaan Menambah Praktikum	35
Tabel 3.15 Spesifikasi Kasus Penggunaan Mengubah Praktikum	36
Tabel 3.16 Spesifikasi Kasus Penggunaan Menghapus Praktikum	38
Tabel 3.17 Spesifikasi Kasus Penggunaan Menambah Tutorial ..	39
Tabel 3.18 Spesifikasi Kasus Penggunaan Mengubah Tutorial ..	40
Tabel 3.19 Spesifikasi Kasus Penggunaan Menghapus Tutorial ..	41

Tabel 3.20 Spesifikasi Kasus Penggunaan Menambah Kategori Tutorial	42
Tabel 3.21 Spesifikasi Kasus Penggunaan Mengubah Kategori Tutorial	42
Tabel 3.22 Spesifikasi Kasus Penggunaan Menghapus Kategori Tutorial	43
Tabel 3.23 Spesifikasi Kasus Penggunaan Menambah Soal Praktikum	45
Tabel 3.24 Spesifikasi Kasus Penggunaan Mengubah Soal Praktikum	46
Tabel 3.25 Spesifikasi Kasus Penggunaan Menghapus Soal Praktikum	47
Tabel 3.26 Spesifikasi Kasus Penggunaan Menambah Soal Tutorial	49
Tabel 3.27 Spesifikasi Kasus Penggunaan Mengubah Soal Tutorial	50
Tabel 3.28 Spesifikasi Kasus Penggunaan Menghapus Soal Tutorial	51
Tabel 3.29 Spesifikasi Kasus Penggunaan Melihat Submissions untuk Asisten	53
Tabel 3.30 Spesifikasi Kasus Penggunaan Melihat Submissions untuk Praktikan.....	54
Tabel 3.31 Spesifikasi Kasus Penggunaan Melihat Scoreboard..	55
Tabel 3.32 Spesifikasi Kasus Penggunaan Menjawab Soal Event	56
Tabel 3.33 Spesifikasi Kasus Penggunaan Mengubah Data Diri Praktikan.....	58
Tabel 3.34 Spesifikasi Kasus Penggunaan Mengubah Password	58
Tabel 3.35 Spesifikasi Kasus Penggunaan Me-reset Password...	61
Tabel 5.1 Kategori Penilaian UAT	109
Tabel 5.2 Kuesioner Pengujian UAT	110
Tabel 5.3 Hasil Perhitungan UAT	111
Tabel 5.4 Hasil Pengujian Grader Versi I.....	112
Tabel 5.5 Hasil Pengujian Grader Versi II	118

DAFTAR KODE SUMBER

Kode Sumber 4.1 Fungsi Memulai Grader.....	79
Kode Sumber 4.2 Fungsi Menghentikan Grader	80
Kode Sumber 4.3 Fungsi Melihat Submissions Praktikum.....	82
Kode Sumber 4.4 Fungsi Melihat Submissions Tutorial.....	83
Kode Sumber 4.5 Fungsi Melihat Scoreboard	85
Kode Sumber 4.6 Fungsi Menjawab Soal Praktikum	86
Kode Sumber 4.7 Fungsi Menjawab Soal Tutorial	87
Kode Sumber 4.8 Fungsi Mengubah Data Diri Praktikan.....	87
Kode Sumber 4.9 Fungsi Mengubah Password.....	88
Kode Sumber 4.10 Fungsi Mengirim Kode Verifikasi ke E-mail	89
Kode Sumber 4.11 Fungsi Verifikasi Reset Password	90
Kode Sumber A.1 Fungsi Menambah Data Praktikan	125
Kode Sumber A.2 Fungsi Mengubah Data Praktikan	125
Kode Sumber A.3 Fungsi Menghapus Data Praktikan.....	125
Kode Sumber A.4 Fungsi Menambah Data Praktikan dengan File Excel.....	126
Kode Sumber A.5 Fungsi Menambah Data Database	127
Kode Sumber A.6 Fungsi Mengubah Data Database	128
Kode Sumber A.7 Fungsi Menghapus Data Database	129
Kode Sumber A.8 Fungsi Menambah Jenis Database.....	129
Kode Sumber A.9 Fungsi Mengubah Jenis Database	129
Kode Sumber A.10 Fungsi Menambah Praktikum.....	130
Kode Sumber A.11 Fungsi Mengubah Praktikum	130
Kode Sumber A.12 Fungsi Menghapus Praktikum	131
Kode Sumber A.13 Fungsi Menambah Tutorial	131
Kode Sumber A.14 Fungsi Mengubah Tutorial	131
Kode Sumber A.15 Fungsi Menghapus Tutorial.....	132
Kode Sumber A.16 Fungsi Menambah Kategori Tutorial	132
Kode Sumber A.17 Fungsi Mengubah Kategori Tutorial	132
Kode Sumber A.18 Fungsi Menghapus Kategori Tutorial.....	132
Kode Sumber A.19 Fungsi Menambah Soal Praktikum	133
Kode Sumber A.20 Fungsi Mengubah Soal Praktikum	133

Kode Sumber A.21 Fungsi Menghapus Soal Praktikum 133

Kode Sumber A.22 Fungsi Menambah Soal Tutorial 133

Kode Sumber A.23 Fungsi Mengubah Soal Tutorial 134

Kode Sumber A.24 Fungsi Menghapus Soal Tutorial..... 134

Kode Sumber A.25 Grader versi I..... 136

Kode Sumber A.26 Grader versi II..... 141

BAB I

PENDAHULUAN

Pada bab ini dijelaskan hal-hal yang menjadi latar belakang, permasalahan yang dihadapi, batasan masalah, tujuan, metodologi dan sistematika penulisan yang digunakan dalam pembuatan buku tugas akhir ini.

1.1 Latar Belakang

Basis data (*database*) adalah kumpulan data, biasanya menjelaskan mengenai aktifitas satu atau lebih organisasi yang terkait. Basis data terdiri dari entitas dan hubungan antar entitas. Basis data biasanya diatur menggunakan sebuah *software* yaitu *Database Management System* (DBMS). Basis data merupakan dasar dari sistem penyimpanan dalam sebuah aplikasi, baik itu aplikasi *desktop*, *website*, maupun *mobile*. Oleh karena itu teori-teori mengenai basis data harus dikuasai oleh semua *developer* perangkat lunak.

Dengan landasan seperti itu, maka hampir di semua jurusan yang berbau Teknologi Informasi mengajarkan pelajaran basis data. Termasuk di Departemen Teknik Informatika ITS. Sistem Basis Data (SBD) adalah mata kuliah di Departemen Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya yang mengajarkan mengenai basis data. Mata kuliah ini mempelajari mengenai dasar-dasar basis data.

Di dalam mata kuliah SBD diajarkan mengenai *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML). DDL merupakan perintah-perintah yang biasa digunakan *administrator* basis data untuk mendefinisikan skema dan subskema basis data. DDL mempunyai fungsi utama untuk mendefinisikan data dalam *database* secara logika. Sedangkan DML merupakan perintah-perintah yang memungkinkan pengguna melakukan akses dan manipulasi data sebagaimana yang telah diorganisasikan sebelumnya dalam

model data yang tepat. DML digunakan untuk memanipulasi basis data yang telah didefinisikan dengan DDL.

Di dalam matakuliah SBD juga terdapat praktikum mengenai materi DDL maupun DML. Penilaian praktikum SBD saat ini masih manual, yaitu praktikan mengeksekusi *query* di DBMS lalu ditunjukkan kepada asistennya untuk dinilai. Sistem praktikum seperti ini tentu sangat tidak efektif, karena membutuhkan waktu lama dan membutuhkan sumber daya asisten yang banyak. Oleh karena itu, diperlukan sebuah aplikasi yang bisa membantu praktikum SBD tersebut. Aplikasi yang akan dibangun adalah Sistem Basis Data *Online judge* (SBDOJ).

SBDOJ sama seperti konsep-konsep *online judge* pada umumnya. SBDOJ merupakan aplikasi berbasis *website*. Aplikasi ini akan digunakan dalam praktikum SBD khususnya pada materi DML yaitu *query* SELECT. Praktikan diharuskan mengumpulkan *query* jawaban dari sebuah soal, kemudian *query* tersebut akan dieksekusi oleh sistem dan akan dinilai keberhasilannya. Dengan adanya aplikasi SBDOJ ini, diharapkan bisa mempermudah asisten maupun praktikan dalam praktikum SBD materi DML.

1.2 Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana cara agar server bisa melakukan proses *judgement* secara cepat?
2. Bagaimana membangun aplikasi berbasis web agar mudah digunakan oleh praktikan dan asisten Sistem Basis Data?

1.3 Batasan Masalah

Permasalahan yang akan diselesaikan dalam tugas akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Aplikasi berbasis web dengan bahasa pemrograman PHP.

2. Aplikasi berbasis kerangka kerja Laravel.
3. Basis data yang digunakan adalah MySQL.
4. *Grader* menggunakan Bahasa pemrograman Python.
5. Sistem digunakan untuk DML *query* SELECT.

1.4 Tujuan

Tujuan dari pengerjaan tugas akhir ini adalah membangun aplikasi yang dapat mempermudah praktikum Sistem Basis Data di Departemen Teknik Informatika ITS pada materi DML *query* SELECT.

1.5 Manfaat

Tugas akhir ini diharapkan dapat memberikan solusi pada praktikum Sistem Basis Data materi DML *query* SELECT di Departemen Teknik Informatika ITS yang selama ini dianggap kurang efektif dengan cara memberikan penilaian secara otomatis kepada *query* yang dikumpulkan praktikan.

1.6 Metodologi

Tahapan-tahapan yang dilakukan dalam pengerjaan tugas akhir ini adalah sebagai berikut:

1. Studi Literatur

Pada tahap ini dilakukan pengumpulan informasi mengenai kerangka kerja Laravel beserta pustaka pendukung lainnya yaitu, bahasa pemrograman PHP, JavaScript, Python dan *database-python connector*. Selain itu juga dilakukan perumusan awal proses *online judge* pada umumnya, terutama bagaimana jalannya sebuah *grader* yang menjadi kunci utama dari *online judge*. Kemudian akan dilakukan pengumpulan informasi mengenai fitur apa saja yang akan diimplementasikan kepada sistem. Informasi dan

pengetahuan yang didapat tersebut digunakan dalam perancangan dan implementasi sistem yang akan dibangun.

2. Analisis dan Perancangan Sistem

Pada tahap ini dilakukan analisis dan perancangan untuk aplikasi Sistem Basis Data Online Judge. Perancangan sistem dilakukan dalam beberapa tahap, yaitu analisis aktor yang terlibat di dalam sistem, analisis kebutuhan fungsional yang kemudian dituangkan pada diagram kasus penggunaan, perancangan arsitektur aplikasi, dan perancangan antarmuka pengguna.

3. Implementasi

Pada tahap ini dilakukan pembuatan elemen aplikasi yang merupakan implementasi yang berpedoman pada rancangan yang telah dibuat pada proses analisis dan perancangan sistem.

4. Pengujian dan evaluasi

Pada tahap ini dilakukan pengujian terhadap elemen aplikasi yang dibangun. Pengujian dan evaluasi perangkat dilakukan untuk mencari masalah yang mungkin timbul, mengevaluasi jalannya program, dan mengadakan perbaikan jika ada kekurangan.

5. Penyusunan buku Tugas Akhir

Pada tahap ini dilakukan proses dokumentasi dan pembuatan laporan dari seluruh konsep, tinjauan pustaka, metode, implementasi, proses yang telah dilakukan, pengujian, evaluasi dan hasil-hasil yang didapatkan selama pengerjaan tugas akhir.

1.7 Sistematika Penulisan

Penulisan buku tugas akhir ini bertujuan untuk mendapatkan gambaran pengerjaan tugas akhir dan diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini:

Bab I Pendahuluan

Bab ini berisi latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat pembuatan tugas akhir, serta metodologi yang digunakan, dan sistematika penulisan buku tugas akhir.

Bab II Tinjauan Pustaka

Bab ini berisi penjelasan mengenai dasar-dasar penunjang yang digunakan untuk mendukung pembuatan tugas akhir ini.

Bab III Analisis dan Perancangan Sistem

Bab ini berisi analisis dan perancangan aplikasi yang akan dibangun. Analisis meliputi deskripsi umum aplikasi, analisis proses bisnis, analisis spesifikasi kebutuhan sistem, dan analisis aktor. Perancangan aplikasi meliputi kasus penggunaan, perancangan arsitektur sistem, dan perancangan antarmuka pada aplikasi.

Bab IV Implementasi

Bab ini berisi hasil penerapan perancangan aplikasi yang telah dibuat pada bab sebelumnya. Implementasi dalam tugas akhir ini adalah dalam bentuk kode sumber beserta penjelasannya dan implementasi antarmuka pengguna pada aplikasi.

Bab V Pengujian dan Evaluasi

Bab ini berisi pengujian dengan metode *black box* untuk mengetahui aspek nilai fungsionalitas dari aplikasi yang dibangun dan nilai kegunaan yang dibuat dengan memperhatikan ketertarikan pada calon pengguna untuk menggunakan aplikasi ini.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari pengerjaan tugas akhir ini dan saran untuk pengembangan aplikasi kedepannya.

Daftar Pustaka

Merupakan daftar referensi yang digunakan dalam tugas akhir ini.

Lampiran

Merupakan bab tambahan yang berisi daftar istilah yang penting pada aplikasi ini dan hasil-hasil dari pengujian aplikasi ini.

BAB II

TINJAUAN PUSTAKA

Pada bab ini dijelaskan pustaka penunjang yang berkaitan dengan pembuatan tugas akhir ini. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap aplikasi dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

2.1 *Online Judge*

Online Judge secara umum adalah server yang berisi deskripsi dari masalah dalam kontes yang berbeda. Serta data yang diatur untuk menilai apakah sebuah solusi tertentu menyelesaikan soal-soal yang berada di server tersebut. [1]

Terdapat beberapa kategori *online judge* yang saat ini telah diimplementasikan, antara lain :

a. *Problem Solving Online Judge*

Online judge ini yang sangat familiar dan sering digunakan oleh para *programmer* untuk mengasah kemampuan logikanya dalam menyelesaikan *problem set* yang disediakan oleh *online judge*. Biasanya *online judge* ini digunakan untuk berlatih *competitive programming*. Contohnya adalah SPOJ, UVA, Toki Learning, dan masih banyak lagi.

b. *HTML & CSS Online Judge*

Online judge untuk HTML dan CSS untuk saat ini masih berupa aplikasi untuk pembelajaran. Untuk itu *online judge* untuk CSS dan HTML terkadang dikenal dengan sebutan *online course* karena tidak ada sistem *rating* untuk tiap soalnya. Sedangkan *online judge* seharusnya memiliki sistem *rating*. Contoh aplikasi *online course* HTML dan CSS adalah Codecademy dan FreeCodeCamp.

c. *SQL Online Judge*

Sama seperti *online judge* untuk HTML dan CSS, *online judge* untuk SQL saat ini masih berupa *online course*. Dikarenakan belum adanya sistem *rating* dalam aplikasi tersebut. Contoh aplikasi *online course* untuk SQL adalah Codecademy dan SqlCourse.

2.2 Database

Basis data (*database*) adalah kumpulan data, biasanya menjelaskan mengenai aktifitas satu atau lebih organisasi yang terkait. Basis data terdiri dari entitas dan hubungan antar entitas. Basis data biasanya diatur menggunakan sebuah *software* yaitu *Database Management System* (DBMS). [2]

2.3 SQL

SQL (*Structured Query Language*) adalah bahasa yang digunakan untuk menggambarkan dan memanipulasi basis data relasional. Standar SQL yang digunakan saat ini adalah SQL-99. Ada 2 aspek pada SQL, yaitu DDL (*Data Definition Language*) dan DML (*Data Manipulation Language*). [2]

2.3.1 Simple Query

Pertanyaan:

Tampilkan NRP, nama, dan Tempat lahir mahasiswa yang mendapatkan nilai E pada FRS urutkan berdasarkan NRP secara *descending*.

Jawaban:

```
SELECT DISTINCT f.nrp, m.nama, m.tempat_lhr
  FROM mahasiswa m, frs f WHERE m.NRP = f.NRP AND
    nilai_huruf = 'E' ORDER BY f.NRP DESC;
```

Hasil:

nrp	nama	tempat_lhr
5203100029	ACHMAD SAFII	Kabupaten Banyuwangi
5203100028	OKTAVIAN EKA PUTRA	Kabupaten Gresik
5203100025	ARDY PRIYONGGO HP	Kabupaten Tulungagung
5203100012	SURYA MUHAMAD K	Kabupaten Bojonegoro
5202100045	MERI RAHMAWATI	Kabupaten Tuban
5202100044	P. DEWANTINING TIANA	Kabupaten Nganjuk
5202100043	WIDYA PRAMUDIYA	Kabupaten Tulungagung
5202100042	YESI NOVIA	Kabupaten Trenggalek
5202100041	YUSRI SM.	Kabupaten Jember
5202100040	ACHMAD GOZALI	Kabupaten Bojonegoro
5202100037	NINO EKA PUTRA WR	Kabupaten Banyuwangi
5202100036	SUYONO	Kabupaten Bondowoso
5202100035	RAHAYU FATNAWATI	Kota Blitar
5202100033	ARIEF KURNIAWAN	Kabupaten Sampang
5202100026	WACHID FAR	Kota Pasuruan
5202100024	ARTO NAFARAIN	Kabupaten Bojonegoro
5202100016	SULAIMAN ARDHianto	Kabupaten Bondowoso
5202100015	HENNY THERESIA K.	Kabupaten Blitar
5202100012	M. IKHWANUL N.A.P.	Kota Malang
5202100010	PANDU PRADANA	Kabupaten Trenggalek
5202100009	GRASIA PRIMARDHIKA	Kabupaten Sampang
5202100006	A. HENDICK SOERJO H.	Kabupaten Bangkalan
5202100005	CHAIRUN NISA	Kabupaten Gresik
5202100004	BAYU SATRIYA N	Kabupaten Lumajang
5202100003	DION KRISTADI L	Kabupaten Lumajang

Gambar 2.1 Hasil Simple Query

2.3.2 Medium Query

Pertanyaan :

Tampilkan semua data mata kuliah beserta nilai hurufnya pada data FRS yang memiliki nilai angka di atas rata-rata nilai angka pada data nilai serta urutkan berdasarkan ID mata kuliah secara *ascending* dan berdasarkan nilai huruf secara *descending*, tanpa ada row yang ganda / terduplikat.

Jawaban:

```
SELECT distinct mk.*, f.nilai_huruf
  FROM matakuliah mk, mk_tawar mkt, frs f
 WHERE mk.ID_MK = mkt.ID_MK
       AND mkt.ID_MK_TAWAR = f.ID_MK_TAWAR
```

```
AND f.NILAI_HURUF IN (SELECT NILAI_HURUF FROM NILAI WHERE
nilai_angka > (SELECT AVG(nilai_angka) FROM nilai))
ORDER BY mk.ID_MK ASC, f.nilai_huruf DESC;
```

Hasil:

ID_MK	MATA_KULIAH	SKS	NILAI_MIN	nilai_huruf
FS1291	FISIKA DASAR I	3	D	BC
FS1291	FISIKA DASAR I	3	D	B
FS1291	FISIKA DASAR I	3	D	AB
FS1291	FISIKA DASAR I	3	D	A
IF1401	PENGANTAR TEKNOLOGI INFORMASI	2	E	BC
IF1401	PENGANTAR TEKNOLOGI INFORMASI	2	E	B
IF1401	PENGANTAR TEKNOLOGI INFORMASI	2	E	AB
IF1401	PENGANTAR TEKNOLOGI INFORMASI	2	E	A
IF1402	ALGORITMA & STRUKTUR DATA	3	E	BC
IF1402	ALGORITMA & STRUKTUR DATA	3	E	B
IF1402	ALGORITMA & STRUKTUR DATA	3	E	AB
IF1402	ALGORITMA & STRUKTUR DATA	3	E	A
IF1403	SISTEM DIGITAL I	2	E	BC
IF1403	SISTEM DIGITAL I	2	E	B
IF1403	SISTEM DIGITAL I	2	E	AB
IF1403	SISTEM DIGITAL I	2	E	A
IF1404	ALJABAR LINIER & MATRIKS	3	C	BC
IF1404	ALJABAR LINIER & MATRIKS	3	C	B
IF1404	ALJABAR LINIER & MATRIKS	3	C	AB
IF1404	ALJABAR LINIER & MATRIKS	3	C	A
IF1405	BAHASA PEMROGRAMAN	3	E	BC
IF1405	BAHASA PEMROGRAMAN	3	E	B
IF1405	BAHASA PEMROGRAMAN	3	E	AB
IF1405	BAHASA PEMROGRAMAN	3	E	A
IF1406	SISTEM DIGITAL II	2	C	BC
IF1406	SISTEM DIGITAL II	2	C	B
IF1406	SISTEM DIGITAL II	2	C	AB
IF1406	SISTEM DIGITAL II	2	C	A

Gambar 2.2 Hasil Medium Query

2.3.3 Complex Query

Pertanyaan:

Tampilkan semua data mahasiswa yang mengambil mata kuliah ARSITEKTUR KOMPUTER yang diajar oleh Ibu Sarwosri, S.Kom urutkan berdasarkan NRP secara *ascending*.

Jawaban:

```
SELECT distinct m.*
FROM mahasiswa m, dosen d, matakuliah mk, mk_tawar mkt,
frs f
WHERE m.NRP = f.NRP
```

```

AND f.ID_MK_TAWAR = mkt.ID_MK_TAWAR
AND mk.ID_MK = mkt.ID_MK
AND d.NIP = mkt.NIP
AND mk.ID_MK in (SELECT ID_MK FROM matakuliah WHERE
mata_kuliah = 'ARSITEKTUR KOMPUTER')
AND d.NIP = (SELECT NIP FROM dosen WHERE nama LIKE
'%Sarwosri%' )
ORDER BY m.NRP ASC

```

Hasil :

NRP	NAMA	JK	TGL_LHR	TEMPAT_LHR	AGAMA
5100100020	RACHEL SHEYNARITA	P	1981-06-13	Kabupaten Madiun	3
5100100056	ICHSAN	L	1981-07-19	Kabupaten Mojokerto	1
5100100064	JANUAR BISAPTANTO	L	1981-07-27	Kabupaten Gresik	1
5100100065	DIYON YUDONO	L	1981-07-28	Kabupaten Nganjuk	1
5100100089	MUHAMMAD MACHMUD	L	1981-08-21	Kabupaten Tuban	1
5101100003	DANI WAFAL FALAH	L	1984-03-04	Kota Blitar	1
5101100012	ADHITYA Y	L	1983-04-11	Kabupaten Ponorogo	1
5101100013	DEDDY IRFANDY	L	1983-02-10	Kabupaten Tulungagung	1
5101100017	SUGENG YUNianto	L	1984-04-03	Kabupaten Sidoarjo	1
5101100021	DWI ARYSSANDHY S	L	1984-05-29	Kota Madiun	1
5101100022	ANTON PRASETYO	L	1984-12-09	Kabupaten Situbondo	1
5101100023	SOALON WALDY R P	L	1983-06-03	Kota Malang	3
5101100024	IMMANUEL AGUSTINUS G	L	1983-11-07	Kabupaten Tulungagung	3
5101100025	AZIZAH	P	1984-11-17	Kota Blitar	1
5101100029	M ARIEF H	L	1984-09-12	Kabupaten Pacitan	1
5101100032	IZAM NURAZWAR	L	1984-11-28	Kabupaten Probolinggo	1
5101100035	WAHYU BUDI SURASTYO	L	1984-07-22	Kabupaten Bondowoso	1
5101100036	SITI KUSTIARSIH	P	1983-07-11	Kabupaten Gresik	1
5101100040	AGUSTINUS WINATA	L	1984-12-03	Kota Probolinggo	2
5101100044	MELATI AQUARISTA I	P	1983-11-08	Kota Blitar	1
5101100048	ARIEF BUDI DARMAWAN	L	1983-10-20	Kota Blitar	1
5101100053	SIGIT PRAMONO	L	1983-11-13	Kota Malang	1
5101100054	EDWIN N A	L	1983-02-27	Kabupaten Magetan	1
5101100055	SATYA YUDISTIRA	L	1983-05-23	Kabupaten Tuban	1
5101100059	RONI SETYO WIBOWO	L	1983-09-09	Kabupaten Sampang	1

Gambar 2.3 Hasil Complex Query

2.4 Kerangka Kerja Laravel

Laravel merupakan *framework* PHP yang menekankan pada kesederhanaan dan fleksibilitas pada desainnya. Laravel dirilis

dibawah lisensi MIT dengan sumber kode yang disediakan di Github. Sama seperti *framework* PHP lainnya, Laravel dibangun dengan basis MVC (*Model-View-Controller*). Laravel dilengkapi *command line tool* yang bernama “Artisan” yang bisa digunakan untuk *packaging bundle* dan instalasi *bundle*. Menurut survey yang dilakukan oleh Sitepoint.com pada Desember 2013 dalam popularitas *framework* PHP, Laravel menduduki urutan teratas. Sehingga menjadikan Laravel sebagai *framework* PHP terbaik untuk tahun 2014. Saat ini Laravel merupakan *framework* dengan versi PHP yang *up-to-date*, karena Laravel menisyaratkan PHP versi 5.3 keatas. [3]

2.5 PHP

PHP (*Hypertext Preprocessor*) adalah bahasa skrip yang dapat diletakkan ke dalam HTML. PHP banyak digunakan untuk memrogram situs web dinamis. PHP memiliki beberapa kelebihan jika dibandingkan dengan bahasa pemrograman yang lain, diantaranya adalah :

- Bahasa pemrograman PHP adalah sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaannya.
- Web Server yang mendukung PHP dapat ditemukan dimana-mana dari mulai Apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
- Dalam sisi pengembangan lebih mudah, karena banyaknya milis-milis dan *developer* yang siap membantu dalam pengembangan.
- Dalam sisi pemahaman, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
- PHP adalah bahasa *open source* yang dapat digunakan berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui console serta juga dapat menjalankan perintah-perintah sistem.

- PHP memiliki delapan tipe data, yaitu: Boolean, Integer, Float/Double, String, Array, Object, Resource, NULL. [4]

2.6 OracleSQL

Oracle adalah *Relational Database Management System* (RDBMS) untuk mengelola informasi secara terbuka, komprehensif dan terintegrasi. Oracle menyediakan solusi yang efisien dan efektif karena kemampuannya.

Kekurangan:

- Merupakan *software* DMBS yang paling mahal, paling rumit, dan paling sulit untuk dipelajari.
- Membutuhkan spesifikasi hardware yang tinggi untuk dapat menjalankan *software* DMBS Oracle supaya berjalan dengan stabil.
- Hanya diperuntukan bagi perusahaan berukuran besar, dan tidak cocok untuk perusahaan kecil maupun menengah.
- Data yang bertambah ukurannya akan mengalami kelambatan proses, jadi harus ada *database management*.

Kelebihan :

- Merupakan *software* DBMS yang handal dan memiliki kemampuan yang tinggi.
- Dapat menangani jumlah data dalam ukuran yang besar.
- Dapat mengolah data dalam ukuran besar dan mengolahnya dengan cepat sehingga didapatkan informasi yang akurat sesuai permintaan pengguna.
- Memiliki kemampuan akan fleksibilitas dan skalabilitas yang dapat memenuhi tuntutan akan data dan informasi yang bervolume besar dan terus-menerus bertambah besar.
- Memiliki kemampuan *technology cluster server*, dimana jika terdapat lebih dari satu unit server misalnya 100 unit server

maka Oracle dapat menjadikan 100 unit server tersebut aktif bekerja bersama sebagai 100 aktif server.

- Memiliki kemampuan untuk manajemen *user* dan tiap *user* bisa diatur hak akses terhadap suatu *database* oleh *database* administrator.
- Bisa berjalan pada lebih dari satu platform sistem operasi.
- Ketika kita mengakses *database* dan kemudian ada kejadian seperti listrik mati misalnya maka data yang sudah kita simpan tidak rusak/hilang. Oracle memiliki kemampuan *flashback*, sehingga semua jenis transaksi yang salah akan dapat dikembalikan, dan dapat menampung data dalam skala besar.
- Dapat bekerja di lingkungan *client/server* (pemrosesan tersebar). [5]

BAB III

ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini menjelaskan tahap analisis dan perancangan dari aplikasi yang akan dibangun pada tugas akhir ini..

3.1 Analisis Sistem

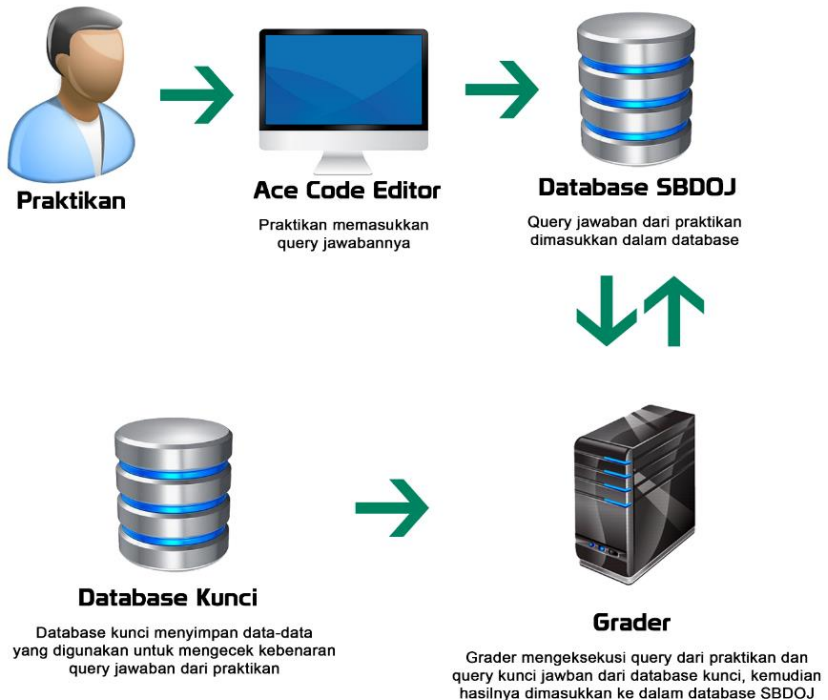
Tahap analisis sistem ini terbagi menjadi beberapa bagian antara lain deskripsi umum perangkat lunak, analisis proses bisnis, spesifikasi kebutuhan sistem, dan analisis aktor.

3.1.1 Deskripsi Umum Aplikasi

Sistem Basis Data Online Judge adalah aplikasi yang ditujukan untuk mempermudah penilaian praktikum Sistem Basis Data di Departemen Teknik Informatika ITS khususnya pada materi DML.

Pada aplikasi ini, asisten bisa memasukkan soal-soal yang digunakan untuk praktikum beserta jawaban dari soal tersebut. Kemudian praktikan akan memasukkan *query* jawaban dari masing-masing soal yang kemudian akan dicek oleh *grader* untuk mengetahui benar atau tidaknya jawaban dari praktikan.

Selain digunakan untuk praktikum, SBDOJ juga bisa digunakan asisten untuk memberi tutorial kepada mahasiswa. Di setiap *event* praktikum, terdapat *scoreboard* untuk melihat rekapitulasi nilai dari mahasiswa. Nilai tersebut antara lain 100 untuk jawaban benar dan tidak melewati batas waktu eksekusi, 0 untuk jawaban salah dan TLE (*Time Limit Exceeded*) untuk jawaban yang melebihi batas waktu eksekusi. Dengan adanya aplikasi SBDOJ ini, asisten akan dimudahkan untuk mendapatkan nilai dari mahasiswa kelasnya dalam praktikum Sistem Basis Data. Deskripsi umum aplikasi dapat dilihat pada Gambar 3.1.



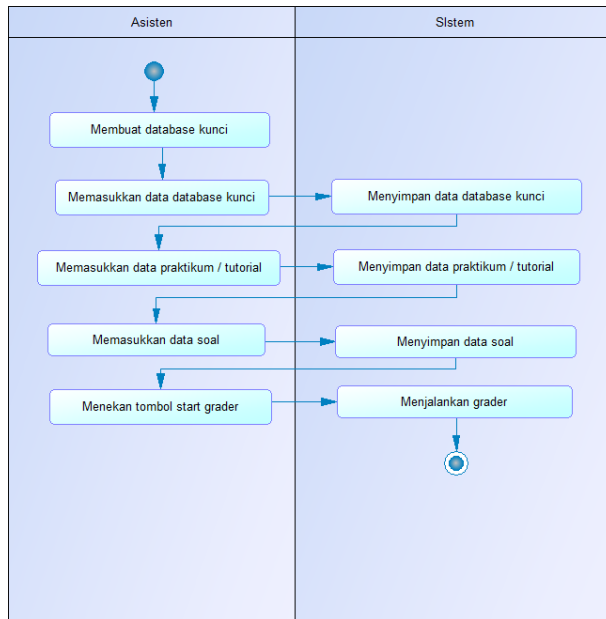
Gambar 3.1 Deskripsi Umum Perangkat Lunak

3.1.2 Analisis Proses Bisnis

Proses bisnis yang digunakan dalam aplikasi ini adalah proses bisnis yang umum terdapat pada *online judge* lainnya. Proses bisnis pada sisi asisten dapat dilihat pada diagram alir pada Gambar 3.2.

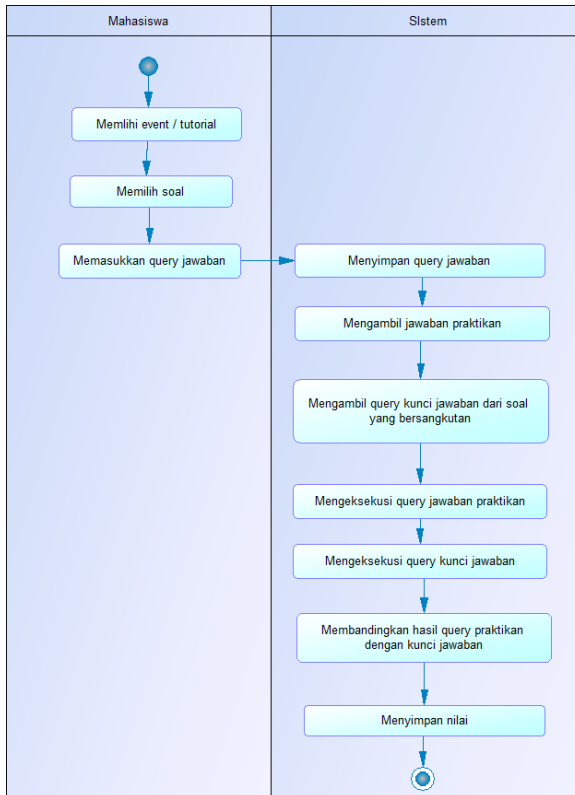
Pada gambar tersebut, asisten harus memasukkan *database* kunci atau *database* yang akan digunakan untuk praktikum. Kemudian asisten harus membuat *event* praktikum atau tutorial. Langkah selanjutnya adalah membuat soal untuk

praktikum tersebut. Soal harus disertai *query* jawaban yang benar dari asisten untuk proses penilaian. Kemudian asisten harus memulai *grader* dari *database* yang digunakan agar penilaian jawaban dari praktikan bisa berlangsung di *background*.



Gambar 3.2 Proses Bisnis Sistem Basis Data *Online judge* pada Asisten

Kemudian, pada Gambar 3.3, dari sisi praktikan dapat memilih *event* praktikum atau tutorial. *Event* praktikum bisa dipilih apabila waktu sudah memasuki waktu mulai praktikum dan belum melewati waktu akhir praktikum. Praktikan kemudian memasukkan *query* jawaban pada soal yang dipilih. Setelah itu *grader* akan mengecek jawaban praktikan dan akan memberikan nilai dari jawaban tersebut. Nilai tersebut yang nantinya akan disimpan pada *database*. Pada aplikasi ini, data praktikan dimasukkan oleh asisten masing-masing kelas. Jadi praktikan tidak bisa mendaftarkan dirinya.



Gambar 3.3 Proses Bisnis Sistem Basis Data *Online judge* pada Praktikan

3.1.3 Analisis Kebutuhan Sistem

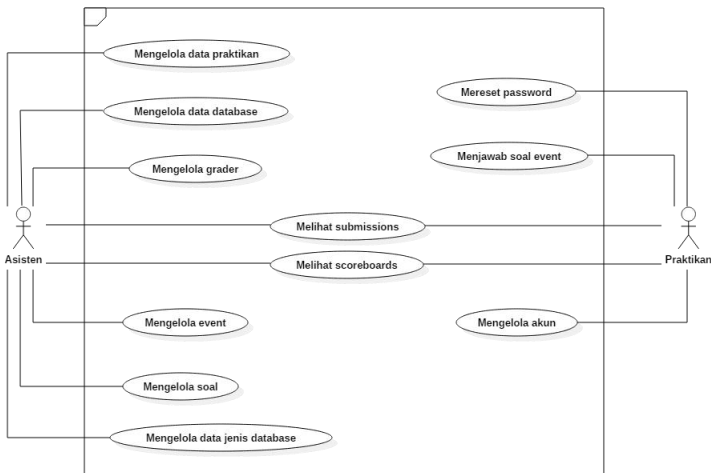
Analisis spesifikasi kebutuhan dalam aplikasi ini mencakup kebutuhan fungsional. Kebutuhan fungsional berisikan proses-proses yang dibutuhkan dalam sistem dan harus dijalankan. Kebutuhan fungsional didapatkan dari analisis proses bisnis yang telah dijelaskan sebelumnya. Kebutuhan fungsional sistem dideskripsikan dalam Tabel 3.1.

Tabel 3.1 Daftar Kebutuhan Fungsional Sistem

Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
F-001	Mengelola data praktikan	Asisten dapat mengelola data praktikan kelasnya masing-masing.
F-002	Mengelola data <i>database</i>	Asisten dapat mengelola <i>database</i> yang digunakan untuk praktikum atau tutorial.
F-003	Mengelola data jenis <i>database</i>	Asisten dapat mengelola jenis <i>database</i> yang digunakan untuk praktikum atau tutorial.
F-004	Mengelola <i>grader</i>	Asisten dapat memulai atau memberhentikan <i>grader</i> tiap-tiap <i>database</i> .
F-005	Mengelola <i>event</i>	Asisten dapat mengelola data praktikum dan tutorial.
F-006	Mengelola soal	Asisten dapat mengelola soal dari praktikum atau tutorial.
F-007	Melihat <i>submissions</i>	Asisten dan praktikan dapat melihat jawaban dari praktikan yang telah di- <i>submit</i> .
F-008	Melihat <i>scoreboards</i>	Asisten dan praktikan dapat melihat <i>scoreboard</i> dari praktikum.
F-009	Menjawab soal <i>event</i>	Praktikan dapat memasukkan <i>query</i> jawaban dari soal-soal praktikum atau tutorial.
F-010	Mengelola akun	Praktikan dapat mengelola akunnya masing-masing.
F-011	Me- <i>reset password</i>	Praktikan bisa melakukan permintaan pengubaan <i>password</i> apabila praktikan lupa <i>password</i> akunnya.

3.1.4 Analisis Aktor

Aktor mendefinisikan pemangku kepentingan yang terlibat dan berinteraksi langsung dengan sistem. Pemangku kepentingan ini bisa berupa manusia maupun sistem atau perangkat lunak yang lain. Pemangku kepentingan yang terdapat pada sistem ini hanya sebagai pengguna. Pengguna yang didefinisikan untuk perangkat lunak ini adalah asisten dan praktikan. Detail tugas dan hak akses pengguna dicantumkan pada Gambar 3.3.



Gambar 3.4 Diagram Kasus Penggunaan

3.2 Perancangan Sistem

Tahap perancangan sistem dibagi menjadi beberapa bagian yaitu kasus penggunaan, perancangan arsitektur aplikasi, dan perancangan antarmuka aplikasi.

3.2.1 Kasus Penggunaan

Kasus penggunaan yang dibutuhkan pada sistem sesuai dengan analisa yang telah dilakukan sebelumnya pada spesifikasi

kebutuhan fungsional dan indentifikasi aktor. Diagram kasus penggunaan dapat dilihat pada Gambar 3.43 dan kode kasus penggunaan pada Tabel 3.2

Tabel 3.2 Keterangan Kode Kasus Penggunaan

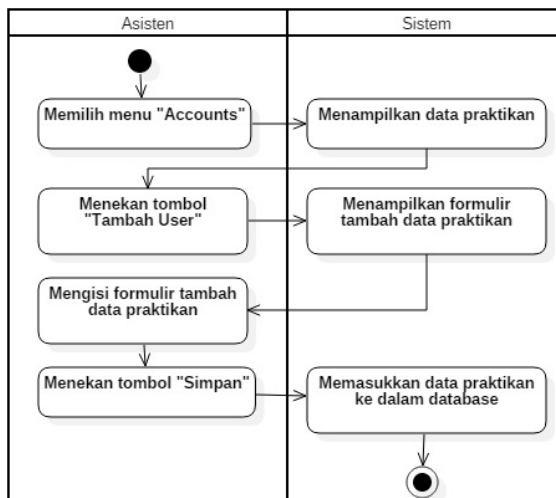
Kode Kasus Penggunaan	Kasus Penggunaan
UC-001	Mengelola data praktikan
UC-002	Mengelola data <i>database</i>
UC-003	Mengelola data jenis <i>database</i>
UC-004	Mengelola <i>grader</i>
UC-005	Mengelola <i>event</i>
UC-006	Mengelola soal
UC-007	Melihat <i>submissions</i>
UC-008	Melihat <i>scoreboards</i>
UC-009	Menjawab soal <i>event</i>
UC-010	Mengelola akun
UC-011	<i>Me-reset password</i>

3.2.1.1 Kasus Penggunaan Mengelola Data Praktikan

Pada kasus penggunaan ini, asisten dapat mengelola data praktikan dari masing-masing kelasnya. Pengelolaan terdiri dari penambahan data praktikan, pengubahan data praktikan dan penghapusan data praktikan. Penambahan data praktikan bisa melalui formulir atau dengan mengunggah *file* Excel.

**Tabel 3.3 Spesifikasi Kasus Penggunaan
Menambah Data Praktikan**

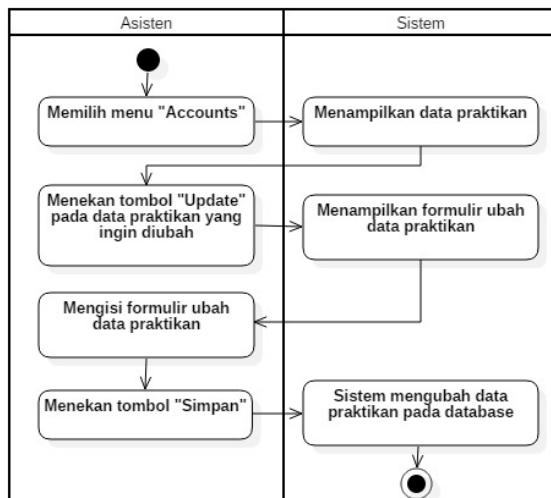
Kode	UC-001.1
Nama	Menambah Data Praktikan
Aktor	Asisten
Deskripsi	Asisten menambah data praktikan kelasnya.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data praktikan ditambahkan ke <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Accounts”. 2. Sistem menampilkan data praktikan. 3. Asisten menekan tombol “Tambah User”. 4. Sistem menampilkan formulir tambah data praktikan. 5. Asisten mengisi formulir tambah data praktikan. 6. Asisten menekan tombol simpan. 7. Sistem memasukkan data praktikan ke dalam <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.5 Diagram Aktivitas Menambah Data Praktikan

**Tabel 3.4 Spesifikasi Kasus Penggunaan
Mengubah Data Praktikan**

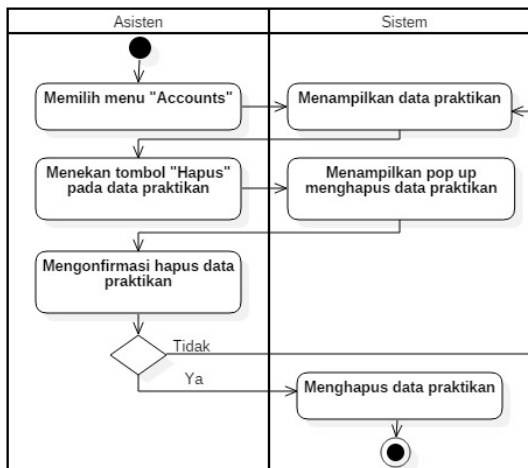
Kode	UC-001.2
Nama	Mengubah Data Praktikan
Aktor	Asisten
Deskripsi	Asisten mengubah data praktikan kelasnya.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data praktikan diubah pada <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “<i>Accounts</i>”. 2. Sistem menampilkan data praktikan. 3. Asisten menekan tombol “<i>Update</i>” pada data praktikan. 4. Sistem menampilkan formulir ubah data praktikan. 5. Asisten mengisi formulir ubah data praktikan. 6. Asisten menekan tombol simpan. 7. Sistem mengubah data praktikan pada <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.6 Diagram Aktivitas Mengubah Data Praktikan

**Tabel 3.5 Spesifikasi Kasus Penggunaan
Menghapus Data Praktikan**

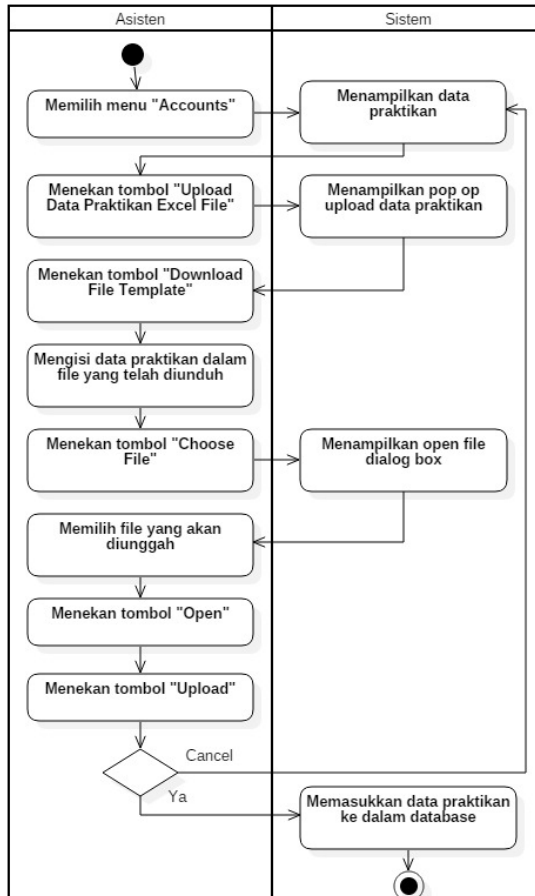
Kode	UC-001.3
Nama	Menghapus Data Praktikan
Aktor	Asisten
Deskripsi	Asisten menghapus data praktikan kelasnya
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data praktikan dihapus dari <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Accounts”. 2. Sistem menampilkan data praktikan. 3. Asisten menekan tombol hapus pada data praktikan. 4. Sistem menampilkan <i>pop-up</i> menghapus data praktikan. 5. Asisten menekan tombol ya. A.5 Asisten menekan tombol tidak. 6. Sistem menghapus data praktikan. 	
Alur Kejadian Alternatif	
A.5. Asisten menekan tombol Tidak.	
A.5.1 Kembali ke Alur Kejadian Normal nomor 2.	



Gambar 3.7 Diagram Aktivitas Meghapus Data Praktikan

**Tabel 3.6 Spesifikasi Kasus Penggunaan
Menambah Data Praktikan dengan *File Excel***

Kode	UC-001.4
Nama	Menambah Data Praktikan dengan <i>File Excel</i>
Aktor	Asisten
Deskripsi	Asisten menambah data praktikan kelasnya dengan mengunggah <i>file Excel</i> .
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data praktikan ditambahkan ke <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> Asisten memilih menu "<i>Accounts</i>". Sistem menampilkan data praktikan. Asisten menekan tombol "<i>Upload Data Praktikan (Excel File)</i>". Sistem menampilkan <i>pop-up Upload Data Praktikan</i>. Asisten mengunduh <i>template file Excel</i> dengan menekan tombol "<i>Download Template File</i>". Asisten mengisi data-data praktikannya ke dalam <i>file</i> yang telah diunduh kemudian disimpan. Asisten menekan tombol "<i>Choose File</i>". Sistem menampilkan <i>pop-up</i> untuk asisten memilih <i>file</i> yang akan diunggah. Asisten menekan tombol "<i>Open</i>". Asisten menekan tombol "<i>Upload</i>". Asisten menekan tombol "<i>Cancel</i>". Sistem memasukkan data praktikan ke dalam <i>database</i>. 	
Alur Kejadian Alternatif	
A.10. Asisten menekan tombol " <i>Cancel</i> ".	
A.10.2 Kembali ke Alur Kejadian Normal nomor 2.	



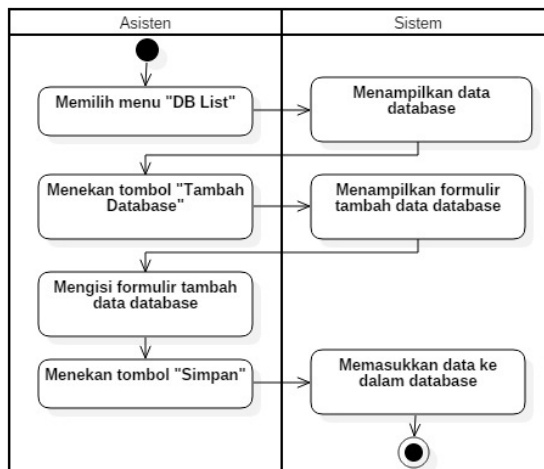
Gambar 3.8 Diagram Aktivitas Menambah Data Praktikan dengan *File Excel*

3.2.1.2 Kasus Penggunaan Mengelola Data *Database*

Pada kasus penggunaan ini, asisten dapat mengelola data *database* yang akan digunakan untuk praktikum atau tutorial. Pengelolaan terdiri dari penambahan data *database*, pengubahan data *database* dan penghapusan data *database*.

**Tabel 3.7 Spesifikasi Kasus Penggunaan
Menambah Data *Database***

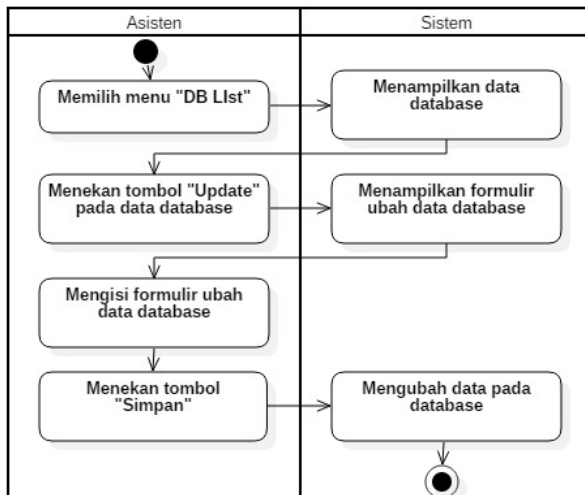
Kode	UC-002.1
Nama	Menambah Data <i>Database</i>
Aktor	Asisten
Deskripsi	Asisten menambah data <i>database</i> untuk praktikum atau tutorial kelasnya.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data <i>database</i> ditambahkan ke <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “DB List”. 2. Sistem menampilkan data <i>database</i>. 3. Asisten menekan tombol “Tambah <i>Database</i>”. 4. Sistem menampilkan formulir tambah data <i>database</i>. 5. Asisten mengisi formulir tambah data <i>database</i>. 6. Asisten menekan tombol simpan. 7. Sistem memasukkan data <i>database</i> ke dalam <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.9 Diagram Aktivitas Menambah Data *Database*

Tabel 3.8 Spesifikasi Kasus Penggunaan
Mengubah Data Database

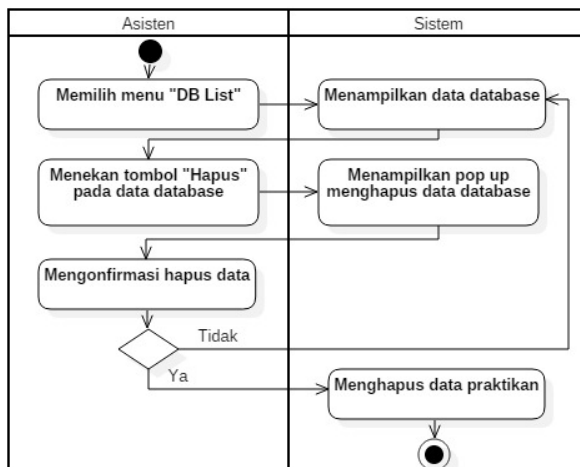
Kode	UC-002.2
Nama	Mengubah Data Database
Aktor	Asisten
Deskripsi	Asisten mengubah data <i>database</i> .
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data <i>database</i> diubah pada <i>database</i> .
Alur Kejadian Normal	
1. Asisten memilih menu “DB List”. 2. Sistem menampilkan data <i>database</i> . 3. Asisten menekan tombol “Update” pada data <i>database</i> . 4. Sistem menampilkan formulir ubah data <i>database</i> . 5. Asisten mengisi formulir ubah data <i>database</i> . 6. Asisten menekan tombol simpan. 7. Sistem mengubah data <i>database</i> pada <i>database</i> .	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.10 Diagram Aktivitas Mengubah Data Database

**Tabel 3.9 Spesifikasi Kasus Penggunaan
Menghapus Data *Database***

Kode	UC-002.3
Nama	Menghapus Data <i>Database</i>
Aktor	Asisten
Deskripsi	Asisten menghapus data <i>database</i> .
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data <i>database</i> telah dihapus dari <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “DB List”. 2. Sistem menampilkan data <i>database</i>. 3. Asisten menekan tombol hapus pada data <i>database</i>. 4. Sistem menampilkan <i>pop-up</i> menghapus data <i>database</i>. 5. Asisten menekan tombol ya. A.5 Asisten menekan tombol tidak. 6. Sistem menghapus data praktikan. 	
Alur Kejadian Alternatif	
A.5. Asisten menekan tombol tidak.	
A.5.1 Kembali ke Alur Kejadian Normal nomor 2.	



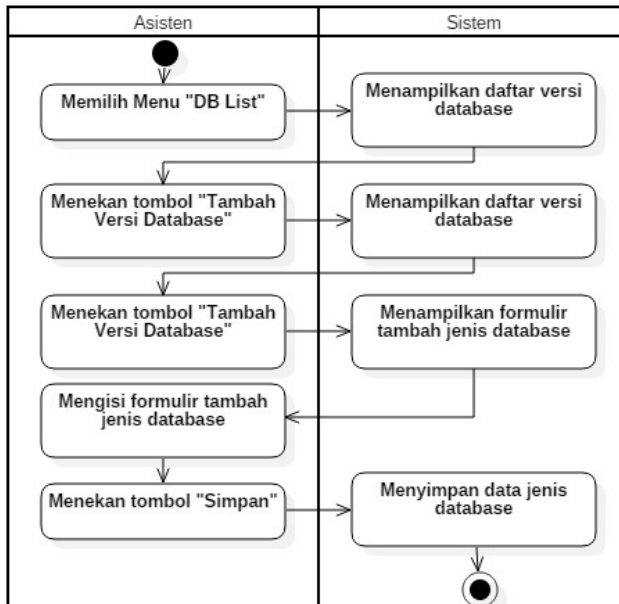
Gambar 3.11 Diagram Aktivitas Menghapus Data *Database*

3.2.1.3 Kasus Penggunaan Mengelola Jenis *Database*

Pada kasus penggunaan ini, asisten dapat mengelola jenis-jenis *database* yang digunakan pada sistem. Jenis *database* antara lain Oracle SQL, MySQL, PostgreSQL, dan lain-lain. Asisten diharuskan melakukan instalasi *plugin* untuk menghubungkan antara Python dengan *database* yang akan digunakan. Pengelolaan terdiri dari penambahan jenis *database* dan perubahan jenis *database*.

**Tabel 3.10 Spesifikasi Kasus Penggunaan
Menambah Jenis *Database***

Kode	UC-003.1
Nama	Menambah Jenis <i>Database</i>
Aktor	Asisten
Deskripsi	Asisten menambah data jenis <i>database</i> .
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Jenis <i>database</i> ditambahkan ke <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “DB <i>List</i>”. 2. Sistem menampilkan daftar <i>database</i>. 3. Pelanggan menekan tombol “Tambah Versi <i>Database</i>”. 4. Sistem menampilkan daftar versi <i>database</i>. 5. Asisten menekan tombol “Tambah Versi <i>Database</i>”. 6. Sistem menampilkan formulir tambah jenis <i>database</i>. 7. Pelanggan mengisi formulir tambah jenis <i>database</i>. 8. Pelanggan menekan tombol simpan. 9. Sistem menyimpan data jenis <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada	



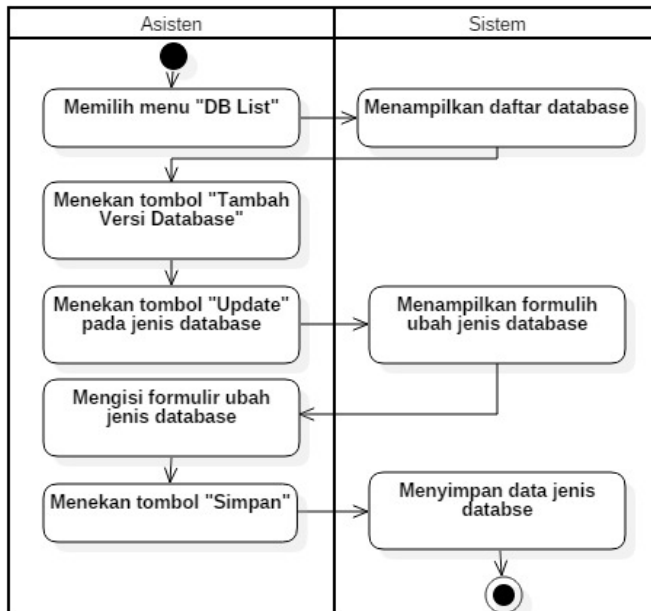
Gambar 3.12 Diagram Aktivitas Menambah Jenis Database

**Tabel 3.11 Spesifikasi Kasus Penggunaan
Mengubah Data Jenis Database**

Kode	UC-003.2
Nama	Mengubah Jenis Database
Aktor	Asisten
Deskripsi	Asisten mengubah data jenis <i>database</i> .
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data jenis <i>database</i> telah diubah.
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Pelanggan memilih menu “DB List”. 2. Sistem menampilkan daftar <i>database</i>. 3. Asisten menekan tombol “Tambah Versi Database”. 	

Tabel 3.11 Spesifikasi Kasus Penggunaan
Mengubah Data Jenis Database

4. Sistem menampilkan daftar versi <i>database</i>
5. Asisten menekan tombol “ <i>Update</i> ” pada jenis <i>database</i> .
6. Sistem menampilkan formulir ubah jenis <i>database</i> .
7. Asisten mengisi formulir ubah jenis <i>database</i> .
8. Asisten menekan tombol simpan.
9. Sistem menyimpan data jenis <i>database</i> .
Alur Kejadian Alternatif
Tidak Ada



Gambar 3.13 Diagram Aktivitas Mengubah Jenis Database

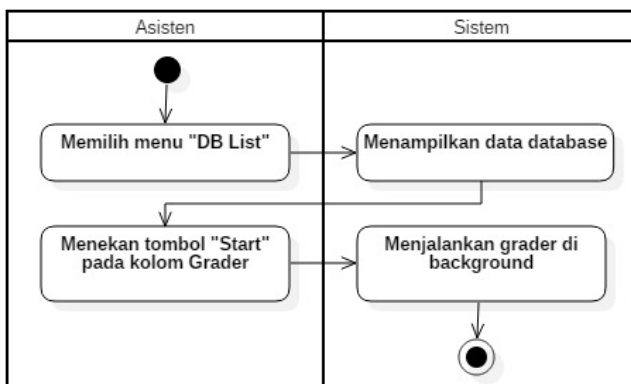
3.2.1.4 Kasus Penggunaan Mengelola Grader

Pada kasus penggunaan ini, asisten dapat mengelola *grader*. *Grader* digunakan untuk mengecek jawaban dari

praktikan dan memberi nilai dari jawaban tersebut. Pengelolaan terdiri dari memulai *grader* dan menghentikan *grader*. *Grader* terdapat 2 jenis, yaitu untuk praktikum dan untuk tutorial.

Tabel 3.12 Spesifikasi Kasus Penggunaan
Memulai Grader

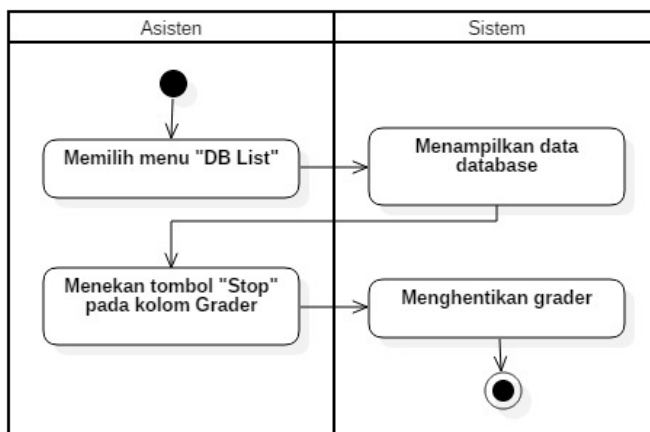
Kode	UC-004.1
Nama	Memulai Grader
Aktor	Asisten
Deskripsi	Asisten menjalankan <i>grader</i> apabila praktikum atau tutorial akan dimulai.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	<i>Grader</i> telah berjalan di <i>background</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “DB List”. 2. Sistem menampilkan data <i>database</i>. 3. Asisten menekan tombol “Start” pada kolom <i>Grader</i> atau <i>Grader Tutorial</i>. 4. Sistem menjalankan <i>grader</i> di <i>background</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.14 Diagram Aktivitas Memulai Grader

**Tabel 3.13 Spesifikasi Kasus Penggunaan
Mengubah Data Praktikan**

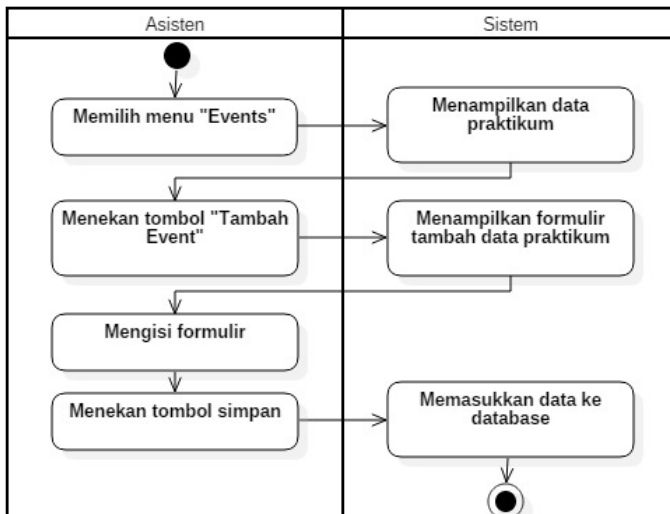
Kode	UC-004.2
Nama	Menghentikan <i>Grader</i>
Aktor	Asisten
Deskripsi	Asisten menghentikan <i>grader</i> apabila praktikum atau tutorial telah selesai.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	<i>Grader</i> telah berhenti.
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “DB List”. 2. Sistem menampilkan data <i>database</i>. 3. Asisten menekan tombol “Stop” pada kolom <i>Grader</i> atau <i>Grader</i> Tutorial. 4. Sistem menghentikan <i>grader</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.15 Diagram Aktivitas Menghentikan *Grader*

3.2.1.5 Kasus Penggunaan Mengelola *Event*

Pada kasus penggunaan ini, asisten dapat mengelola data *event*. *Event* terdiri dari praktikum dan tutorial. Pengelolaan terdiri dari menambah praktikum, mengubah praktikum, menghapus praktikum, menambah tutorial, mengubah tutorial, menghapus tutorial, menambah kategori tutorial, mengubah kategori tutorial, dan menghapus kategori tutorial.



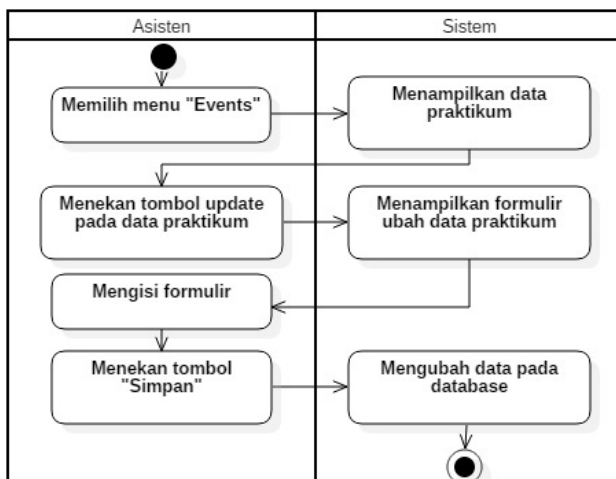
Gambar 3.16 Diagram Aktivitas Menambah Praktikum

Tabel 3.14 Spesifikasi Kasus Penggunaan Menambah Praktikum

Kode	UC-005.1
Nama	Menambah Praktikum
Aktor	Asisten
Deskripsi	Asisten menambah data praktikum kelasnya
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data praktikum ditambahkan ke <i>database</i> .

**Tabel 3.14 Spesifikasi Kasus Penggunaan
Menambah Praktikum**

Alur Kejadian Normal
<ol style="list-style-type: none"> 1. Asisten memilih menu “<i>Events</i>”. 2. Sistem menampilkan data praktikum. 3. Asisten menekan tombol “Tambah <i>Event</i>”. 4. Sistem menampilkan formulir tambah data praktikum. 5. Asisten mengisi formulir tambah data praktikum. 6. Asisten menekan tombol simpan. 7. Sistem memasukkan data praktikum ke dalam <i>database</i>.
Alur Kejadian Alternatif
Tidak ada.



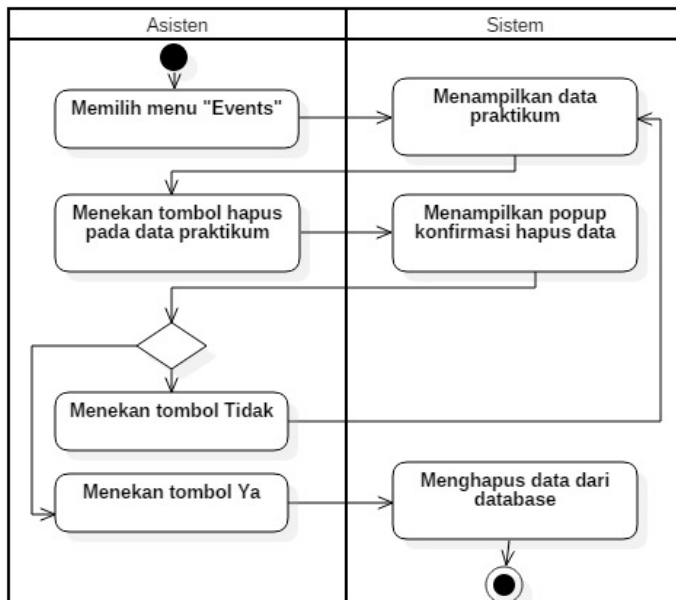
Gambar 3.17 Diagram Aktivitas Mengubah Praktikum

**Tabel 3.15 Spesifikasi Kasus Penggunaan
Mengubah Praktikum**

Kode	UC-005.2
Nama	Mengubah Praktikum
Aktor	Asisten
Deskripsi	Asisten mengubah data praktikum.

**Tabel 3.15 Spesifikasi Kasus Penggunaan
Mengubah Praktikum**

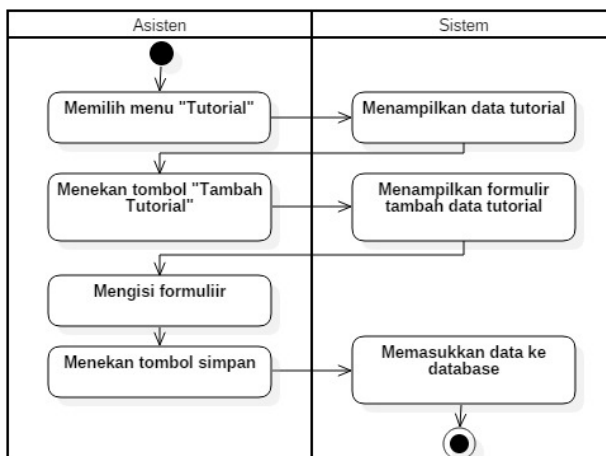
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data praktikum diubah pada <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “<i>Events</i>”. 2. Sistem menampilkan data praktikum. 3. Asisten menekan tombol “<i>Update</i>” pada data praktikum. 4. Sistem menampilkan formulir ubah data praktikum. 5. Asisten mengisi formulir ubah data praktikum. 6. Asisten menekan tombol simpan. 7. Sistem mengubah data praktikum pada <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.18 Diagram Aktivitas Menghapus Praktikum

**Tabel 3.16 Spesifikasi Kasus Penggunaan
Menghapus Praktikum**

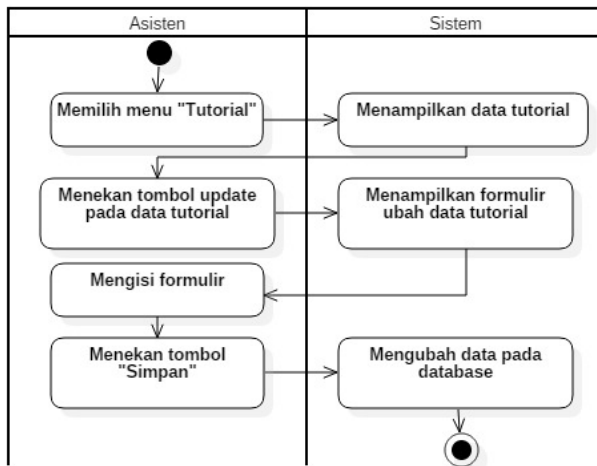
Kode	UC-005.3
Nama	Menghapus Praktikum
Aktor	Asisten
Deskripsi	Asisten menghapus data praktikum.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data praktikum dihapus dari <i>database</i> .
Alur Kejadian Normal	
1. Asisten memilih menu “Events”. 2. Sistem menampilkan data praktikum. 3. Asisten menekan tombol hapus pada praktikum. 4. Sistem menampilkan <i>pop-up</i> menghapus data praktikum. 5. Asisten menekan tombol ya. A.5 Asisten menekan tombol tidak. 6. Sistem menghapus data praktikum.	
Alur Kejadian Alternatif	
A.5. Asisten menekan tombol tidak.	
A.5.2 Kembali ke Alur Kejadian Normal nomor 2.	



Gambar 3.19 Diagram Aktivitas Menambah Tutorial

**Tabel 3.17 Spesifikasi Kasus Penggunaan
Menambah Tutorial**

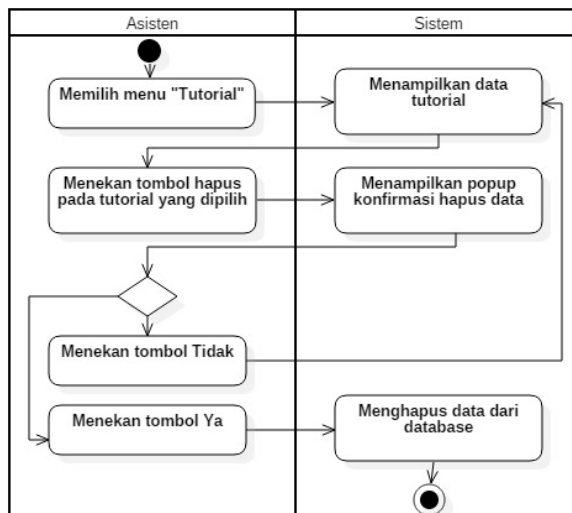
Kode	UC-005.4
Nama	Menambah Tutorial
Aktor	Asisten
Deskripsi	Asisten menambah data tutorial kelasnya.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data tutorial ditambahkan ke <i>database</i>
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten menekan tombol “Tambah Tutorial”. 4. Sistem menampilkan formulir tambah data tutorial. 5. Asisten mengisi formulir tambah data tutorial. 6. Asisten menekan tombol simpan. 7. Sistem memasukkan data tutorial ke dalam <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.20 Diagram Aktivitas Mengubah Tutorial

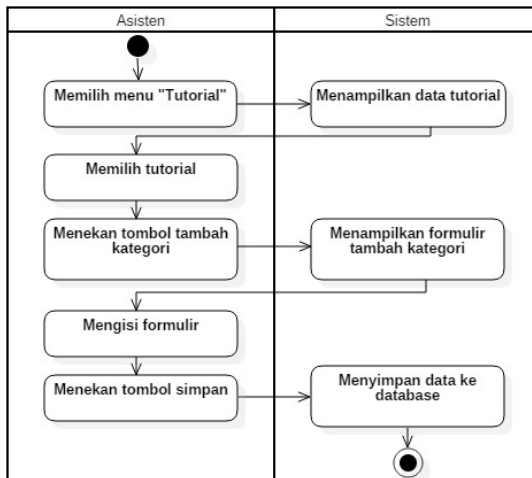
Tabel 3.18 Spesifikasi Kasus Penggunaan Mengubah Tutorial

Kode	UC-005.5
Nama	Mengubah Tutorial
Aktor	Asisten
Deskripsi	Asisten mengubah data tutorial.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data tutorial diubah pada <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten menekan tombol “<i>Update</i>” pada data tutorial. 4. Sistem menampilkan formulir ubah data tutorial. 5. Asisten mengisi formulir ubah data tutorial. 6. Asisten menekan tombol simpan. 7. Sistem mengubah data tutorial pada <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	

**Gambar 3.21 Diagram Aktivitas Menghapus Tutorial**

Tabel 3.19 Spesifikasi Kasus Penggunaan Menghapus Tutorial

Kode	UC-005.6
Nama	Menghapus Tutorial
Aktor	Asisten
Deskripsi	Asisten menghapus data tutorial.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data tutorial dihapus dari <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten menekan tombol hapus pada data tutorial. 4. Sistem menampilkan <i>pop-up</i> menghapus data tutorial. 5. Asisten menekan tombol ya. 6. Sistem menghapus data tutorial. 	
Alur Kejadian Alternatif	
A.5. Asisten menekan tombol tidak.	
A.5.2 Kembali ke Alur Kejadian Normal nomor 2.	

**Gambar 3.22 Diagram Aktivitas Menambah Kategori Tutorial**

**Tabel 3.20 Spesifikasi Kasus Penggunaan
Menambah Kategori Tutorial**

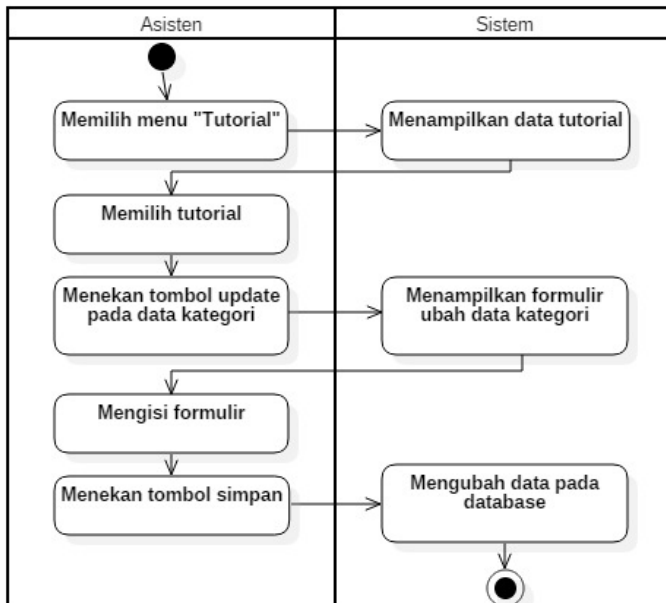
Kode	UC-005.7
Nama	Menambah Kategori Tutorial
Aktor	Asisten
Deskripsi	Asisten menambah data kategori untuk tutorial kelasnya.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data kategori ditambahkan ke <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten memilih salah satu tutorial. 4. Asisten menekan tombol “Tambah Kategori”. 5. Sistem menampilkan formulir tambah data kategori tutorial. 6. Asisten mengisi formulir tambah data kategori tutorial. 7. Asisten menekan tombol simpan. 8. Sistem memasukkan data kategori tutorial dalam <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	

**Tabel 3.21 Spesifikasi Kasus Penggunaan
Mengubah Kategori Tutorial**

Kode	UC-005.8
Nama	Mengubah Kategori Tutorial
Aktor	Asisten
Deskripsi	Asisten mengubah data kategori tutorial
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data kategori tutorial diubah pada <i>database</i>
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten memilih salah satu tutorial. 4. Asisten menekan tombol “Update” pada data kategori tutorial 	

**Tabel 3.21 Spesifikasi Kasus Penggunaan
Mengubah Kategori Tutorial**

5. Sistem menampilkan formulir ubah data kategori tutorial.
6. Asisten mengisi formulir ubah data kategori tutorial.
7. Asisten menekan tombol simpan.
8. Sistem mengubah data kategori tutorial pada <i>database</i> .
Alur Kejadian Alternatif
Tidak ada.

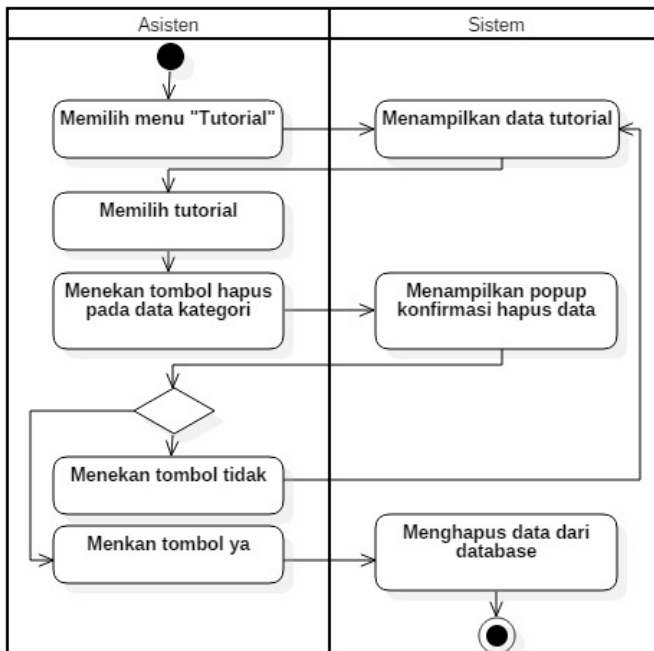


Gambar 3.23 Diagram Aktivitas Mengubah Kategori Tutorial

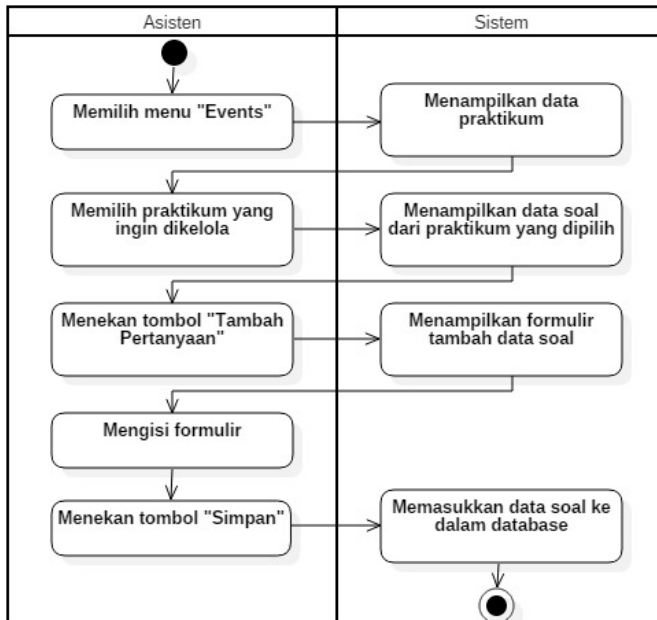
**Tabel 3.22 Spesifikasi Kasus Penggunaan
Menghapus Kategori Tutorial**

Kode	UC-005.9
Nama	Menghapus Kategori Tutorial
Aktor	Asisten
Deskripsi	Asisten menghapus data kategori tutorial.
Tipe	Fungsional

Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data kategori tutorial dihapus dari <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten memilih salah satu tutorial. 4. Asisten menekan tombol hapus pada data kategori tutorial. 5. Sistem menampilkan <i>pop-up</i> konfirmasi penghapusan. 6. Asisten menekan tombol ya. 7. Sistem menghapus data kategori tutorial. 	
Alur Kejadian Alternatif	
A.6. Asisten menekan tombol tidak.	
A.6.2 Kembali ke Alur Kejadian Normal nomor 2.	



Gambar 3.24 Diagram Aktivitas Menghapus Kategori Tutorial



Gambar 3.25 Diagram Aktivitas Menambah Soal Praktikum

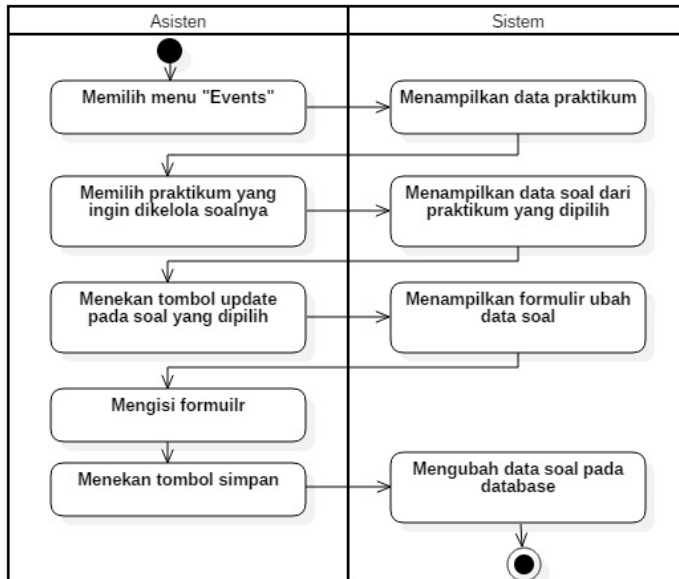
**Tabel 3.23 Spesifikasi Kasus Penggunaan
Menambah Soal Praktikum**

Kode	UC-006.1
Nama	Menambah Soal Praktikum
Aktor	Asisten
Deskripsi	Asisten menambah data soal untuk praktikum kelasnya.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data soal praktikum ditambahkan ke <i>database</i> .
Alur Kejadian Normal	
1. Asisten memilih menu “ <i>Events</i> ”. 2. Sistem menampilkan data praktikum. 3. Asisten memilih praktikum yang ingin dikelola soalnya.	

4. Sistem menampilkan data soal dari praktikum yang dipilih.
5. Asisten menekan tombol “Tambah Pertanyaan”.
6. Sistem menampilkan formulir tambah data soal praktikum.
7. Asisten mengisi formulir tambah data soal praktikum.
8. Asisten menekan tombol simpan.
9. Sistem memasukkan data soal praktikum ke dalam <i>database</i> .
Alur Kejadian Alternatif
Tidak ada.

**Tabel 3.24 Spesifikasi Kasus Penggunaan
Mengubah Soal Praktikum**

Kode	UC-006.2
Nama	Mengubah Soal Praktikum
Aktor	Asisten
Deskripsi	Asisten mengubah data soal praktikum.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data soal praktikum diubah pada <i>database</i> .
Alur Kejadian Normal	
1.	Asisten memilih menu “ <i>Events</i> ”.
2.	Sistem menampilkan data praktikum.
3.	Asisten memilih praktikum yang ingin dikelola soalnya.
4.	Sistem menampilkan data soal dari praktikum yang dipilih.
5.	Asisten menekan tombol “ <i>Update</i> ” pada soal yang dipilih.
6.	Sistem menampilkan formulir ubah data soal praktikum.
7.	Asisten mengisi formulir ubah data soal praktikum.
8.	Asisten menekan tombol simpan.
9.	Sistem mengubah data soal praktikum pada <i>database</i> .
Alur Kejadian Alternatif	
Tidak ada.	



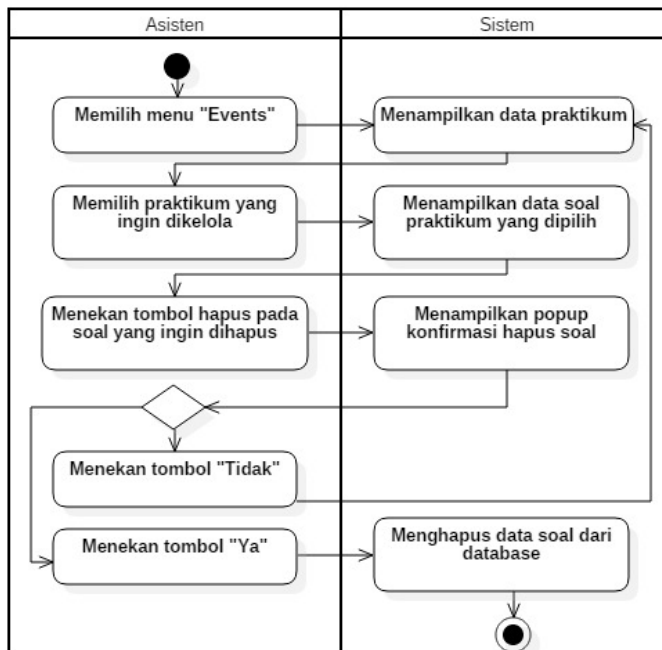
Gambar 3.26 Diagram Aktivitas Mengubah Soal Praktikum

**Tabel 3.25 Spesifikasi Kasus Penggunaan
Menghapus Soal Praktikum**

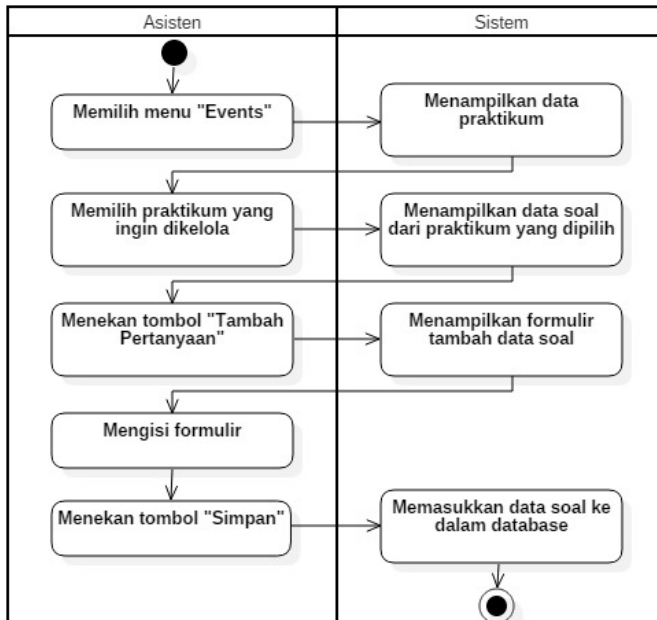
Kode	UC-006.3
Nama	Menghapus Soal Praktikum
Aktor	Asisten
Deskripsi	Asisten menghapus data soal praktikum.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data soal praktikum dihapus dari <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “<i>Events</i>”. 2. Sistem menampilkan data praktikum. 3. Asisten memilih praktikum yang ingin dikelola soalnya. 4. Sistem menampilkan data soal dari praktikum yang dipilih 	

**Tabel 3.25 Spesifikasi Kasus Penggunaan
Menghapus Soal Praktikum**

5. Asisten menekan tombol hapus pada soal yang ingin dihapus.
6. Sistem menampilkan <i>pop-up</i> menghapus data soal praktikum.
7. Asisten menekan tombol ya.
A.7 Asisten menekan tombol tidak.
8. Sistem menghapus data soal praktikum.
Alur Kejadian Alternatif
A.7. Asisten menekan tombol tidak.
A.7.2 Kembali ke Alur Kejadian Normal nomor 2.



Gambar 3.27 Diagram Aktivitas Menghapus Soal Praktikum



Gambar 3.28 Diagram Aktivitas Menambah Soal Tutorial

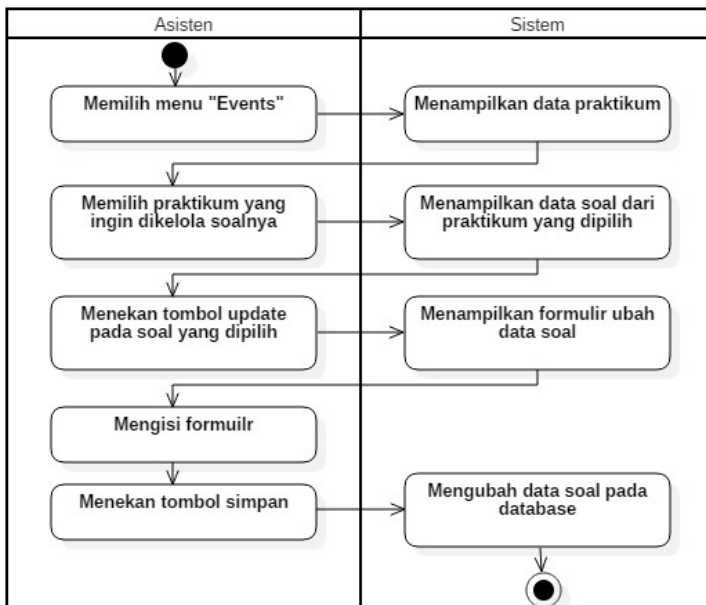
**Tabel 3.26 Spesifikasi Kasus Penggunaan
Menambah Soal Tutorial**

Kode	UC-006.4
Nama	Menambah Soal Tutorial
Aktor	Asisten
Deskripsi	Asisten menambah data soal tutorial.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data tutorial ditambahkan ke <i>database</i> .
Alur Kejadian Normal	
1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten memilih tutorial yang ingin dikelola soalnya. 4. Sistem menampilkan data soal dari tutorial yang dipilih.	

5. Asisten menekan tombol “Tambah Pertanyaan”.
6. Sistem menampilkan formulir tambah data soal tutorial.
7. Asisten mengisi formulir tambah data soal tutorial.
8. Asisten menekan tombol simpan.
9. Sistem memasukkan data soal tutorial ke dalam *database*.

Alur Kejadian Alternatif

Tidak ada.



Gambar 3.29 Diagram Aktivitas Mengubah Soal Tutorial

Tabel 3.27 Spesifikasi Kasus Penggunaan Mengubah Soal Tutorial

Kode	UC-006.5
Nama	Mengubah Soal Tutorial
Aktor	Asisten
Deskripsi	Asisten mengubah data soal tutorial.
Tipe	Fungsional

**Tabel 3.27 Spesifikasi Kasus Penggunaan
Mengubah Soal Tutorial**

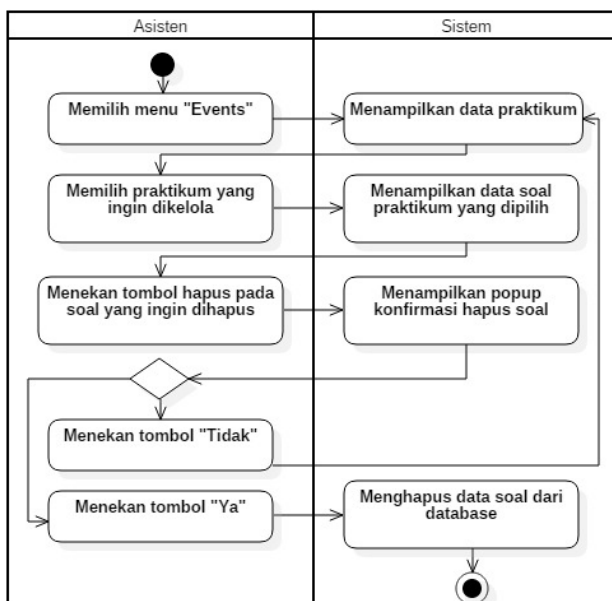
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data soal tutorial telah diubah pada <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten memilih tutorial yang ingin dikelola soalnya. 4. Sistem menampilkan data soal dari tutorial yang dipilih. 5. Asisten menekan tombol “<i>Update</i>” pada soal yang dipilih. 6. Sistem menampilkan formulir ubah data soal tutorial. 7. Asisten mengisi formulir ubah data soal tutorial. 8. Asisten menekan tombol simpan. 9. Sistem mengubah data soal tutorial pada <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada.	

**Tabel 3.28 Spesifikasi Kasus Penggunaan
Menghapus Soal Tutorial**

Kode	UC-006.6
Nama	Menghapus Soal Tutorial
Aktor	Asisten
Deskripsi	Asisten menghapus data soal tutorial.
Tipe	Fungsional
Kondisi Awal	Asisten telah <i>login</i> ke dalam aplikasi.
Kondisi Akhir	Data soal tutorial telah dihapus pada <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “Tutorial”. 2. Sistem menampilkan data tutorial. 3. Asisten memilih tutorial yang ingin dikelola soalnya. 4. Sistem menampilkan data soal dari tutorial yang dipilih. 5. Asisten menekan tombol hapus pada soal tutorial yang ingin dihapus. 	

**Tabel 3.28 Spesifikasi Kasus Penggunaan
Menghapus Soal Tutorial**

6. Sistem menampilkan <i>pop-up</i> konfirmasi menghapus data soal tutorial.
7. Asisten menekan tombol ya. A.7 Asisten menekan tombol tidak.
8. Sistem menghapus data soal tutorial.
Alur Kejadian Alternatif
A.7. Asisten menekan tombol tidak.
A.7.2 Kembali ke Alur Kejadian Normal nomor 2.



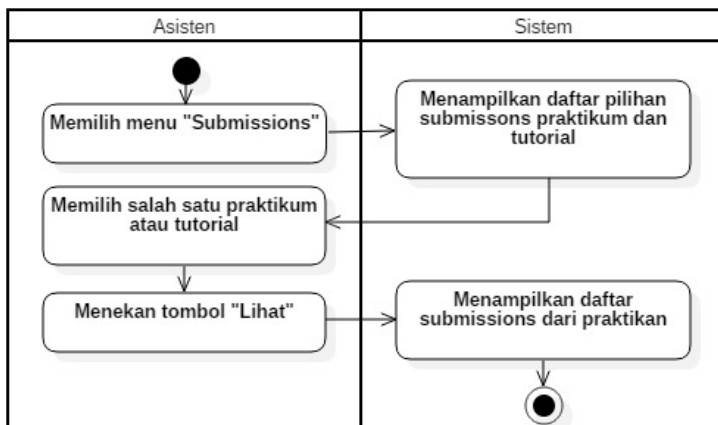
Gambar 3.30 Diagram Aktivitas Menghapus Soal Tutorial

3.2.1.6 Kasus Penggunaan Melihat *Submissions*

Pada kasus penggunaan ini, asisten dan praktikan dapat melihat jawaban-jawaban yang telah di-*submit* pada praktikum atau tutorial tertentu.

**Tabel 3.29 Spesifikasi Kasus Penggunaan
Melihat *Submissions* untuk Asisten**

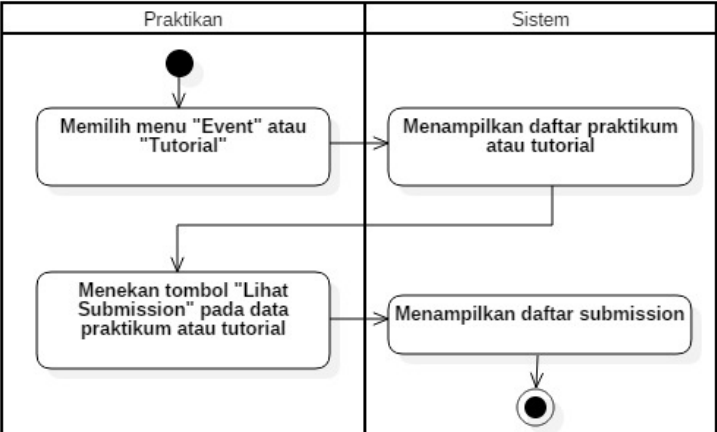
Kode	UC-007.1
Nama	Melihat <i>Submissions</i> untuk Asisten
Aktor	Asisten
Deskripsi	Asisten melihat jawaban yang telah di- <i>submit</i> oleh praktikan.
Tipe	Fungsional
Kondisi Awal	Asisten atau praktikan telah <i>login</i> .
Kondisi Akhir	Daftar <i>submissions</i> ditampilkan.
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten memilih menu “<i>Submissions</i>”. 2. Sistem menampilkan daftar pilihan <i>submissions</i> praktikum dan tutorial. 3. Asisten memilih praktikum atau tutorial. 4. Asisten menekan tombol “Lihat”. 5. Sistem menampilkan daftar <i>submissions</i> dari praktikum atau tutorial yang dipilih. 	
Alur Kejadian Alternatif	
Tidak ada	



Gambar 3.31 Diagram Aktivitas Melihat *Submissions* untuk Asisten

**Tabel 3.30 Spesifikasi Kasus Penggunaan
Melihat *Submissions* untuk Praktikan**

Kode	UC-007.2
Nama	Melihat <i>Submissions</i> untuk Praktikan
Aktor	Praktikan
Deskripsi	Praktikan melihat jawaban yang telah di- <i>submit</i> olehnya.
Tipe	Fungsional
Kondisi Awal	Praktikan telah <i>login</i> .
Kondisi Akhir	Daftar <i>submissions</i> ditampilkan.
Alur Kejadian Normal	
<ol style="list-style-type: none">1. Praktikan memilih menu “<i>Events</i>” atau “Tutorial”.2. Sistem menampilkan daftar praktikum atau tutorial.3. Praktikan menekan tombol “Lihat <i>submission</i>” pada praktikum atau tutorial yang dipilih.4. Sistem menampilkan daftar <i>submissions</i> dari praktikum atau tutorial yang dipilih.	
Alur Kejadian Alternatif	
Tidak Ada	



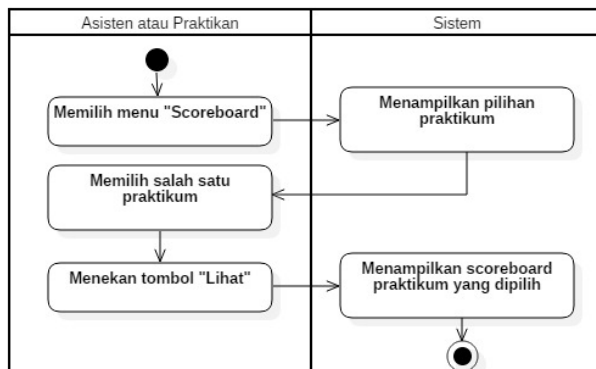
Gambar 3.32 Diagram Aktivitas Melihat *Submissions* untuk Praktikan

3.2.1.7 Kasus Penggunaan Melihat *Scoreboard*

Pada kasus penggunaan ini, asisten dan praktikan bisa melihat *scoreboard* dari suatu praktikum. *Scoreboard* tidak tersedia untuk tutorial.

Tabel 3.31 Spesifikasi Kasus Penggunaan Melihat *Scoreboard*

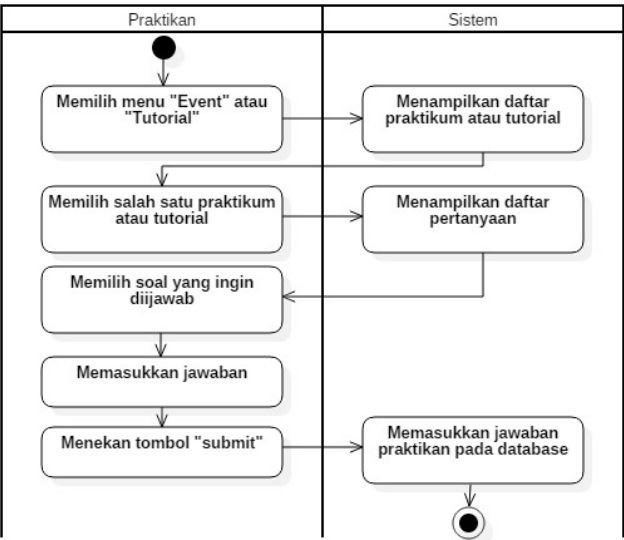
Kode	UC-008
Nama	Melihat <i>Scoreboard</i>
Aktor	Asisten dan Praktikan
Deskripsi	Asisten atau praktikan bisa melihat <i>scoreboard</i> dari suatu praktikum.
Tipe	Fungsional
Kondisi Awal	Asisten atau praktikan telah <i>login</i> .
Kondisi Akhir	<i>Scoreboard</i> suatu praktikum ditampilkan.
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Asisten atau praktikan memilih menu “<i>Scoreboard</i>”. 2. Sistem menampilkan pilihan praktikum. 3. Asisten atau praktikan memilih praktikum yang ingin dilihat <i>scoreboard</i>-nya.. 4. Asisten atau praktikan menekan tombol “Lihat”. 5. Sistem menampilkan <i>scoreboard</i> dari praktikum yang dipilih. 	
Alur Kejadian Alternatif	
Tidak ada.	



Gambar 3.33 Diagram Aktivitas Melihat *Scoreboard*

3.2.1.8 Kasus Penggunaan Menjawab Soal *Event*

Pada kasus penggunaan ini, praktikan dapat menjawab soal dari praktikum dan tutorial kelasnya masing-masing. Praktikan hanya bisa menjawab soal selama *event* tersebut masih aktif.



Gambar 3.34 Diagram Aktivitas Menjawab Soal *Event*

Tabel 3.32 Spesifikasi Kasus Penggunaan Menjawab Soal *Event*

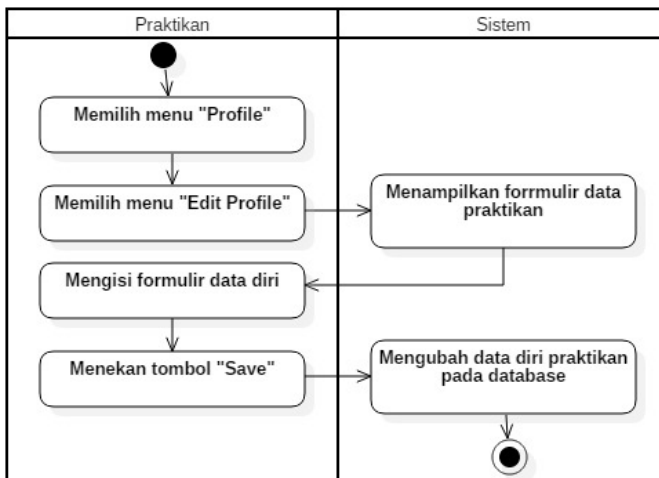
Kode	UC-009
Nama	Menjawab Soal <i>Event</i>
Aktor	Praktikan
Deskripsi	Praktikan dapat menjawab soal-soal pada praktikum atau tutorial.
Tipe	Fungsional
Kondisi Awal	Praktikan telah <i>login</i> .
Kondisi Akhir	Jawaban dari praktikan telah masuk pada <i>database</i> .

**Tabel 3.32 Spesifikasi Kasus Penggunaan
Menjawab Soal Event**

Alur Kejadian Normal
<ol style="list-style-type: none"> 1. Praktikan memilih menu “Events” atau “Tutorial”. 2. Sistem menampilkan daftar praktikum atau tutorial. 3. Praktikan memilih praktikum atau tutorial. 4. Sistem menampilkan daftar pertanyaan dari praktikum atau tutorial yang dipilih. 5. Praktikan memilih soal yang ingin dijawab. 6. Praktikan memasukkan jawaban. 7. Praktikan menekan tombol “Submit”. 8. Sistem memasukkan jawaban pada <i>database</i>.
Alur Kejadian Alternatif
Tidak ada.

3.2.1.9 Kasus Penggunaan Mengelola Akun

Pada kasus penggunaan ini, praktikan dapat mengelola akunnya masing-masing. Pengelolaan akun terdiri dari mengubah data diri praktikan dan mengganti *password* dari akunnya.



Gambar 3.35 Diagram Aktivitas Mengubah Data Diri Praktikan

**Tabel 3.33 Spesifikasi Kasus Penggunaan
Mengubah Data Diri Praktikan**

Kode	UC-010.1
Nama	Mengubah Data Diri Praktikan
Aktor	Praktikan
Deskripsi	Praktikan dapat mengubah data diri dari akunnya.
Tipe	Fungsional
Kondisi Awal	Praktikan telah <i>login</i> .
Kondisi Akhir	Data diri praktikan telah diubah pada <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Praktikan memilih menu “<i>Profile</i>”. 2. Praktikan memilih menu “<i>Edit Profile</i>” 3. Sistem menampilkan formulir data diri praktikan. 4. Praktikan mengisi formulir data diri. 5. Praktikan menekan tombol “<i>Save</i>”. 6. Sistem mengubah data diri praktikan pada <i>database</i>. 	
Alur Kejadian Alternatif	
Tidak ada	

Tabel 3.34 Spesifikasi Kasus Penggunaan Mengubah *Password*

Kode	UC-010.2
Nama	Mengubah <i>Password</i>
Aktor	Praktikan
Deskripsi	Praktikan dapat mengubah <i>password</i> dari akunnya.
Tipe	Fungsional
Kondisi Awal	Praktikan telah <i>login</i> .
Kondisi Akhir	<i>Password</i> akun praktikan diubah di <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Praktikan memilih menu “<i>Profile</i>”. 2. Praktikan memilih menu “<i>Edit Profile</i>” 3. Sistem menampilkan formulir data diri praktikan. 4. Praktikan memilih menu “<i>Change Password</i>”. 	

Tabel 3.34 Spesifikasi Kasus Penggunaan Mengubah Password

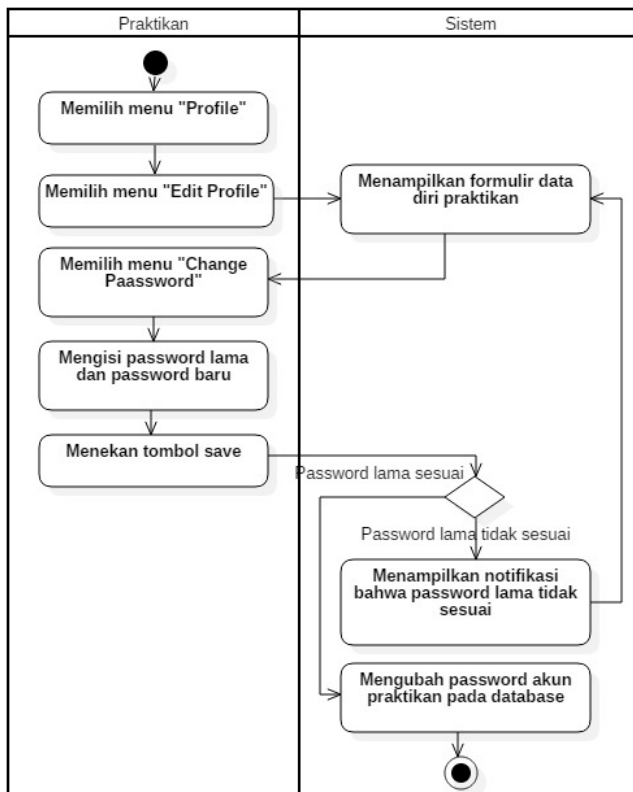
5. Praktikan memasukkan *password* lama dan *password* baru.
6. Praktikan menekan tombol “Save”.
- A.6 *Password* lama tidak sesuai
7. Sistem mengubah *password* akun praktikan pada *database*.

Alur Kejadian Alternatif

A.6. *Password* lama tidak sesuai.

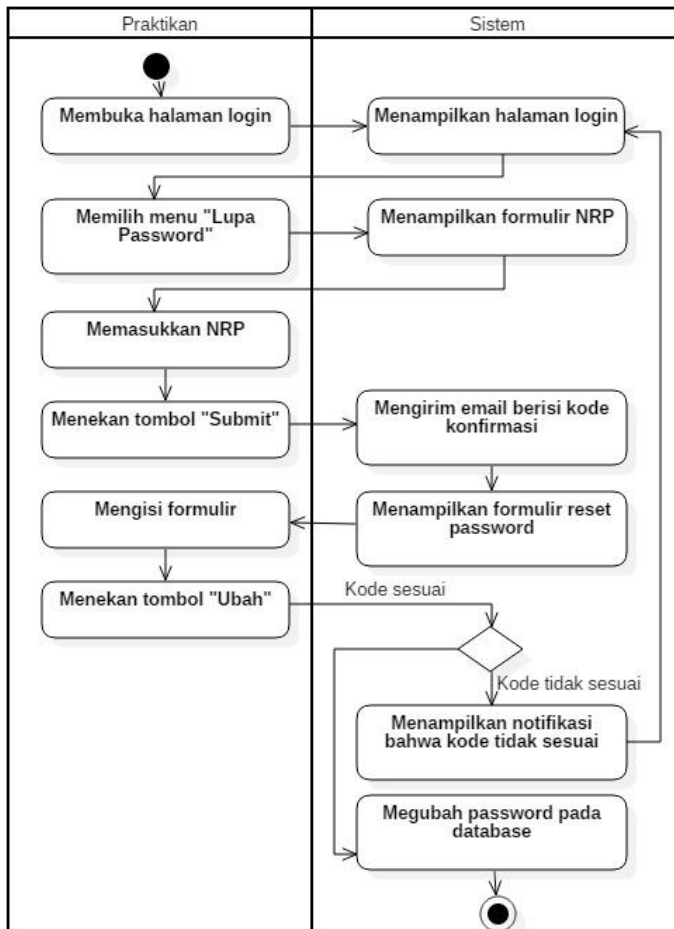
A.6.2 Sistem menampilkan notifikasi bahwa *password* lama tidak sesuai.

A.6.3 Kembali ke Alur Kejadian Normal nomor 3

**Gambar 3.36 Diagram Aktivitas Mengubah Password**

3.2.1.10 Kasus Penggunaan Me-reset Password

Pada kasus penggunaan ini, praktikan dapat mengubah *password*-nya apabila praktikan lupa *password* dari akun tersebut. Kode dari *reset password* akan dikirim via *e-mail* dan akan dikirim ke *e-mail* praktikan tersebut.



Gambar 3.37 Diagram Aktivitas Me-reset Password

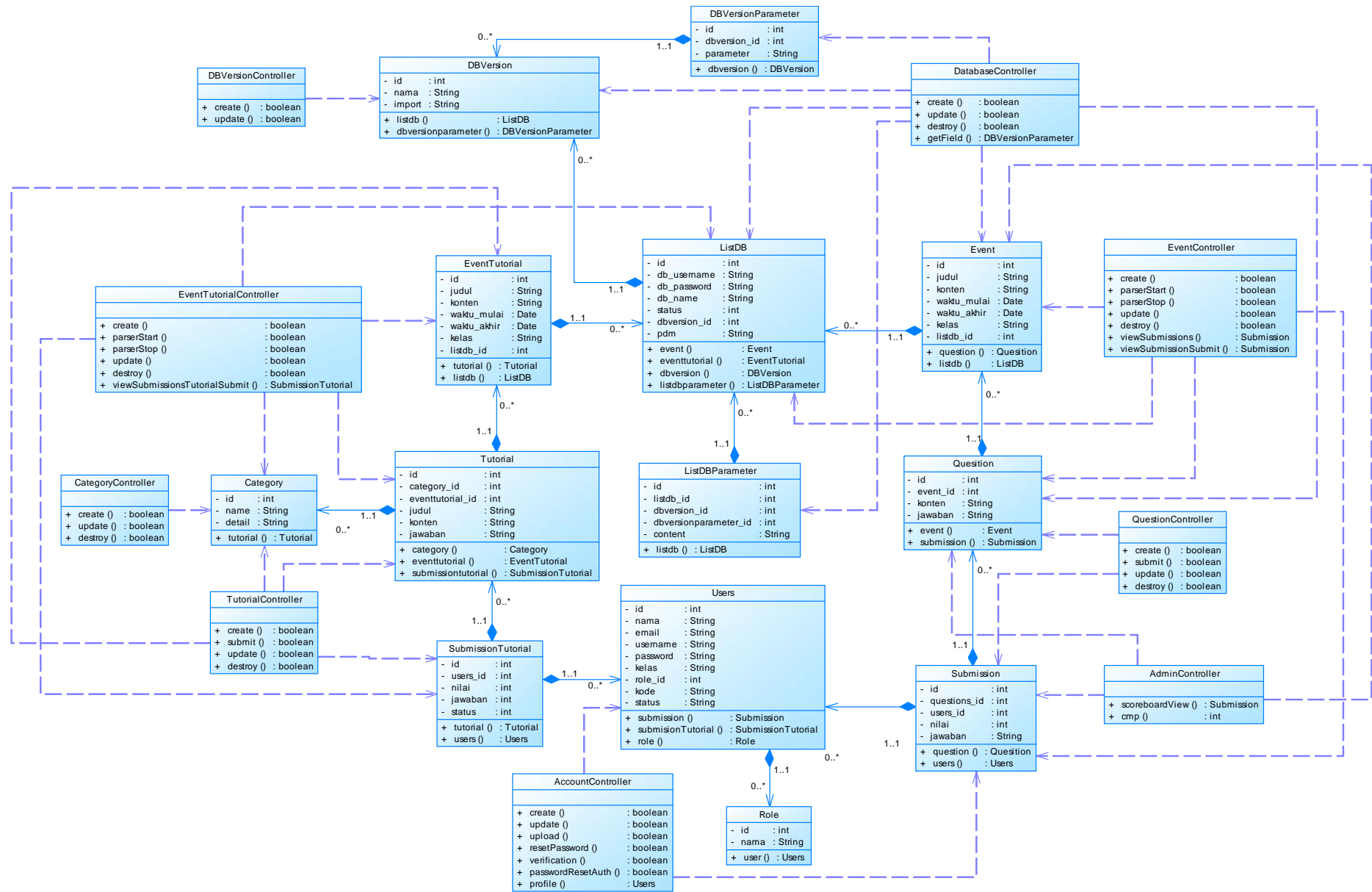
Tabel 3.35 Spesifikasi Kasus Penggunaan Me-reset Password

Kode	UC-011
Nama	Me-reset Password
Aktor	Praktikan
Deskripsi	Praktikan me-reset <i>password</i> akunnya.
Tipe	Fungsional
Kondisi Awal	Praktikan lupa <i>password</i> dari akunnya.
Kondisi Akhir	<i>Password</i> akun praktikan diubah di <i>database</i> .
Alur Kejadian Normal	
<ol style="list-style-type: none"> 1. Praktikan membuka halaman <i>login</i> dari sistem. 2. Sistem menampilkan halaman <i>login</i>. 3. Praktikan memilih menu “Lupa <i>password</i>”. 4. Sistem menampilkan formulir untuk praktikan memasukkan NRP. 5. Praktikan memasukkan NRP. 6. Praktikan menekan tombol “<i>Submit</i>”. 7. Sistem mengirim <i>e-mail</i> yang berisi kode untuk konfirmasi <i>reset password</i>. 8. Sistem menampilkan formulir <i>reset password</i>. 9. Praktikan mengisi formulir dan kode yang telah dikirim. 10. Praktikan menekan tombol “Ubah”. A.10. Kode <i>reset password</i> tidak sesuai. 11. Sistem mengubah <i>password</i> akun praktikan pada <i>database</i>. 	
Alur Kejadian Alternatif	
A.10. Kode <i>reset password</i> tidak sesuai.	
A.10.1 Sistem menampilkan notifikasi bahwa kode yang dimasukkan tidak sesuai.	
A.10.2 Kembali ke Alur Kejadian Normal nomor 2.	

3.2.2 Perancangan Diagram Kelas

Pada Gambar 3.38 menunjukkan diagram kelas dari aplikasi SBDOJ. Aplikasi SBDOJ merupakan aplikasi berbasis web dan menggunakan *framework* Laravel. Oleh Karena itu, diagram kelas dibuat berdasarkan konsep MVC

[Halaman ini sengaja dikosongkan]



Gambar 3.38 Diagram Kelas SBDOJ

[Halaman ini sengaja dikosongkan]

3.2.2.1 Kelas Role

Role	
- id	: int
- nama	: String
+ user ()	: Users

Gambar 3.39 Kelas Role

Kelas Role merupakan model untuk tabel role, yang menyimpan data-data peran dari pengguna SBDOJ. Tabel role hanya berisi 3 data, yaitu Admin, Asisten dan Praktikan. Kelas Role memiliki relasi *zero-to-many* dengan kelas Users.

3.2.2.2 Kelas Users

Users	
- id	: int
- nama	: String
- email	: String
- username	: String
- password	: String
- kelas	: String
- role_id	: int
- kode	: String
- status	: String
+ submission ()	: Submission
+ submissionTutorial ()	: SubmissionTutorial
+ role ()	: Role

Gambar 3.40 Kelas Users

Kelas Users merupakan model untuk tabel users, yang menyimpan data-data dari praktikan maupun asisten. Kelas Users juga digunakan untuk *authentication*. Kelas Users memiliki relasi *zero-to-many* dengan kelas Submission, relasi *zero-to-many* dengan kelas SubmissionTutorial dan relasi *one-to-one* dengan kelas Role.

3.2.2.3 Kelas DBVersionParameter

DBVersionParameter	
- id	: int
- dbversion_id	: int
- parameter	: String
+ dbversion ()	: DBVersion

Gambar 3.41 Kelas DBVersionParameter

Kelas DBVersionParameter merupakan model untuk tabel dbversionparameter, yang menyimpan data-data dari parameter tiap jenis *database*. Kelas DBVersionParameter memiliki relasi *one-to-one* dengan kelas DBVersion.

3.2.2.4 Kelas DBVersion

DBVersion	
- id	: int
- nama	: String
- import	: String
+ listdb ()	: ListDB
+ dbversionparameter ()	: DBVersionParameter

Gambar 3.42 Kelas DBVersion

Kelas DBVersion merupakan model untuk tabel dbversion, yang menyimpan data-data dari jenis-jenis *database* yang dapat digunakan. Kelas DBVersion memiliki relasi *zero-to-many* dengan kelas DBVersionParameter dan relasi *zero-to-many* dengan kelas ListDB.

3.2.2.5 Kelas ListDBParameter

ListDBParameter	
- id	: int
- listdb_id	: int
- dbversion_id	: int
- dbversionparameter_id	: int
- content	: String
+ listdb ()	: ListDB

Gambar 3.43 Kelas ListDParamter

Kelas `ListDBParameter` merupakan model untuk tabel `listdbparameter`, yang menyimpan data-data parameter dari tiap-tiap *database* yang dapat digunakan, misal *username*, *password* dan *host* dari masing-masing *database*. Kelas `ListDBParameter` memiliki relasi *one-to-one* dengan kelas `ListDB`.

3.2.2.6 Kelas ListDB

ListDB	
- id	: int
- db_username	: String
- db_password	: String
- db_name	: String
- status	: int
- dbversion_id	: int
- pdm	: String
+ event ()	: Event
+ eventtutorial ()	: EventTutorial
+ dbversion ()	: DBVersion
+ listdbparameter ()	: ListDBParameter

Gambar 3.44 Kelas ListDB

Kelas `ListDB` merupakan model untuk tabel `listdb`, yang menyimpan data-data dari *database* yang dapat digunakan pada praktikum dan tutorial. Kelas `ListDB` memiliki relasi *one-to-one* dengan kelas `DBVersion`, relasi *zero-to-many* dengan kelas `Event` dan `EventTutorial` dan relasi *zero-to-many* dengan kelas `ListDBParameter`.

3.2.2.7 Kelas Event

Event	
- id	: int
- judul	: String
- konten	: String
- waktu_mulai	: Date
- waktu_akhir	: Date
- kelas	: String
- listdb_id	: int
+ question ()	: Quesition
+ listdb ()	: ListDB

Gambar 3.45 Kelas Event

Kelas Event merupakan model untuk tabel event, yang menyimpan data-data praktikum dari semua kelas. Kelas Event memiliki relasi *one-to-one* dengan kelas ListDB dan relasi *zero-to-many* dengan kelas Question.

3.2.2.8 Kelas EventTutorial

EventTutorial	
- id	: int
- judul	: String
- konten	: String
- waktu_mulai	: Date
- waktu_akhir	: Date
- kelas	: String
- listdb_id	: int
+ tutorial ()	: Tutorial
+ listdb ()	: ListDB

Gambar 3.46 Kelas EventTutorial

Kelas EventTutorial merupakan model untuk tabel eventtutorial, yang menyimpan data-data tutorial dari semua kelas. Kelas EventTutorial memiliki relasi *one-to-one* dengan kelas ListDB dan relasi *zero-to-many* dengan kelas Tutorial.

3.2.2.9 Kelas Question

Question	
- id	: int
- event_id	: int
- konten	: String
- jawaban	: String
+ event ()	: Event
+ submission ()	: Submission

Gambar 3.47 Kelas Question

Kelas Question merupakan model untuk tabel question, yang menyimpan data-data pertanyaan dan jawaban dari suatu

praktikum. Kelas Question memiliki relasi *one-to-one* dengan kelas Event dan relasi *zero-to-many* dengan kelas Submission.

3.2.2.10 Kelas Tutorial

Tutorial	
- id	: int
- category_id	: int
- eventtutorial_id	: int
- judul	: String
- konten	: String
- jawaban	: String
+ category ()	: Category
+ eventtutorial ()	: EventTutorial
+ submissiontutorial ()	: SubmissionTutorial

Gambar 3.48 Kelas Tutorial

Kelas Tutorial merupakan model untuk tabel tutorial, yang menyimpan data-data pertanyaan dan jawaban dari suatu tutorial. Kelas Tutorial memiliki relasi *one-to-one* dengan kelas Category, relasi *one-to-one* dengan kelas EventTutorial dan relasi *zero-to-many* dengan kelas SubmissionTutorial.

3.2.2.11 Kelas Category

Category	
- id	: int
- name	: String
- detail	: String
+ tutorial ()	: Tutorial

Gambar 3.49 Kelas Category

Kelas Category merupakan model untuk tabel category, yang menyimpan data-data kategori dari pertanyaan-pertanyaan tutorial. Kelas Category memiliki relasi *zero-to-many* dengan kelas Tutorial.

3.2.2.12 Kelas Submission

Submission	
- id	: int
- questions_id	: int
- users_id	: int
- nilai	: int
- jawaban	: String
+ question ()	: Question
+ users ()	: Users

Gambar 3.50 Kelas Submission

Kelas Submission merupakan model untuk tabel submission, yang menyimpan jawaban yang dikumpulkan oleh praktikan pada praktikum. Tabel submission juga menyimpan nilai dari jawaban tersebut setelah dinilai oleh *grader*. Kelas Submission memiliki relasi *one-to-one* dengan kelas Question dan Users.

3.2.2.13 Kelas SubmissionTutorial

SubmissionTutorial	
- id	: int
- users_id	: int
- nilai	: int
- jawaban	: int
- status	: int
+ tutorial ()	: Tutorial
+ users ()	: Users

Gambar 3.51 Kelas SubmissionTutorial

Kelas SubmissionTutorial merupakan model untuk tabel submissiontutorial, yang menyimpan jawaban yang dikumpulkan oleh praktikan pada tutorial. Kelas SubmissionTutorial memiliki relasi *one-to-one* dengan kelas Tutorial dan Users.

3.2.2.14 Kelas AccountController

AccountController	
+ create ()	: boolean
+ update ()	: boolean
+ upload ()	: boolean
+ resetPassword ()	: boolean
+ verification ()	: boolean
+ passwordResetAuth ()	: boolean
+ profile ()	: Users

Gambar 3.52 Kelas AccountController

Kelas AccountController merupakan *controller* yang mengelola data-data pengguna SBDOJ. Kelas ini juga menangani proses *reset password* apabila pengguna lupa *password*-nya. AccountController memiliki *dependency* dengan kelas Users dan Submission.

3.2.2.15 Kelas DBVersionController

DBVersionController	
+ create ()	: boolean
+ update ()	: boolean

Gambar 3.53 Kelas DBVersionController

Kelas DBVersionController merupakan *controller* yang mengelola data-data jenis-jeniss *database* yang dapat digunakan pada SBDOJ. AccountController memiliki *dependency* dengan kelas DBVersion.

3.2.2.16 Kelas DatabaseController

DatabaseController	
+ create ()	: boolean
+ update ()	: boolean
+ destroy ()	: boolean
+ getField ()	: DBVersionParameter

Gambar 3.54 Kelas DatabaseController

Kelas DatabaseController merupakan *controller* yang mengelola data-data *database* yang dapat digunakan pada praktikum dan tutorial. DatabaseController memiliki *dependency* dengan kelas DBVersion, ListDB, DBVersionParameter, ListDBParameter, Event dan Question.

3.2.2.17 Kelas EventController

EventController	
+ create ()	: boolean
+ parserStart ()	: boolean
+ parserStop ()	: boolean
+ update ()	: boolean
+ destroy ()	: boolean
+ viewSubmissions ()	: Submission
+ viewSubmissionSubmit ()	: Submission

Gambar 3.55 Kelas EventController

Kelas EventController merupakan *controller* yang mengelola data-data praktikum. *Controller* ini juga mengelola proses memulai dan menghentikan *grader*. EventController memiliki *dependency* dengan kelas ListDB, Event, Question dan Submission.

3.2.2.18 Kelas QuestionController

QuestionController	
+ create ()	: boolean
+ submit ()	: boolean
+ update ()	: boolean
+ destroy ()	: boolean

Gambar 3.56 Kelas QuestionController

Kelas QuestionController merupakan *controller* yang mengelola data-data pertanyaan untuk praktikum. *Controller* ini juga mengelola proses pengumpulan jawaban praktikum dari

praktikan. QuestionController memiliki *dependency* dengan kelas Question dan Submission.

3.2.2.19 Kelas EventTutorialController

EventTutorialController	
+ create ()	: boolean
+ parserStart ()	: boolean
+ parserStop ()	: boolean
+ update ()	: boolean
+ destroy ()	: boolean
+ viewSubmissionsTutorialSubmit ()	: SubmissionTutorial

Gambar 3.57 Kelas EventTutorialController

Kelas EventTutorialController merupakan *controller* yang mengelola data-data tutorial. *Controller* ini juga mengelola proses memulai dan menghentikan *grader* untuk tutorial. EventTutorialController memiliki *dependency* dengan kelas ListDB, EventTutorial, Tutorial, SubmissionTutorial dan Category.

3.2.2.20 Kelas CategoryController

CategoryController	
+ create ()	: boolean
+ update ()	: boolean
+ destroy ()	: boolean

Gambar 3.58 Kelas CategoryController

Kelas CategoryController merupakan *controller* yang mengelola data-data kategori tutorial. Kelas ini memiliki *dependency* dengan kelas Category.

3.2.2.21 Kelas TutorialController

TutorialController	
+ create ()	: boolean
+ submit ()	: boolean
+ update ()	: boolean
+ destroy ()	: boolean

Gambar 3.59 Kelas TutorialController

Kelas TutorialController merupakan *controller* yang mengelola data-data pertanyaan untuk tutorial. *Controller* ini juga mengelola proses pengumpulan jawaban tutorial dari praktikan. TutorialController memiliki *dependency* dengan kelas Tutorial, SubmissionTutorial, Category dan EventTutorial.

3.2.2.22 Kelas AdminController

AdminController	
+ scoreboardView ()	: Submission
+ cmp ()	: int

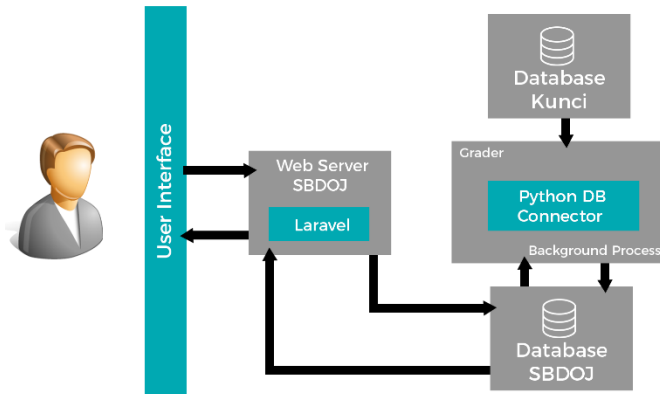
Gambar 3.60 Kelas AdminController

Kelas AdminController merupakan *controller* yang mengelola proses untuk melihat *scoreboard* dari suatu praktikum. Dalam kelas ini terdapat fungsi untuk membandingkan nilai sehingga *scoreboard* dapat terurut berdasarkan nilai dan waktu pengumpulan jawaban praktikan. AdminController memiliki *dependency* dengan kelas Event, Question dan Submission.

3.2.3 Perancangan Arsitektur Aplikasi

Aplikasi yang dibangun adalah aplikasi berbasis web. Oleh karena itu dibutuhkan web server, *grader*, *database* SBDOJ

dan *database* kunci untuk memenuhi kebutuhan sistem. *Grader* berjalan pada *background process*.



Gambar 3.61 Arsitektur Aplikasi

[Halaman sengaja dikosongkan]

BAB IV IMPLEMENTASI

Pada bab ini, dijelaskan mengenai implementasi perangkat lunak sesuai analisis dan perancangan proses bisnis secara umum, pada SBDOJ yang dijabarkan pada bab sebelumnya. Implementasi yang dijelaskan meliputi lingkungan pembangunan perangkat lunak, kode sumber utama dan implementasi antarmuka perangkat lunak.

4.1 Lingkungan Implementasi

Dalam merancang perangkat lunak ini, digunakan beberapa perangkat pendukung yang terdiri dari perangkat keras dan perangkat lunak.

4.1.1 Lingkungan Implementasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam membangun aplikasi SBDOJ adalah sebagai berikut:

Jenis	: <i>Personal Computer</i>
Tipe	: LENOVO 10132
Prosesor	: Intel® Core™ i3-4150 CPU @ 3.50GHz
Memori	: 4096MB RAM
Sistem Operasi	: Windows 10 Pro 64-bit

4.1.2 Lingkungan Implementasi Perangkat Lunak

Perangkat lunak yang digunakan dalam membangun aplikasi SBDOJ adalah sebagai berikut:

- Sublime Text 3 digunakan untuk menyunting kode.
- StarUML 2.7.0 digunakan untuk membuat diagram kasus penggunaan, diagram aktivitas dan diagram kelas.
- Oracle SQL dan MySQL digunakan untuk antarmuka pengelola basis data.
- Apache digunakan untuk web server.

4.2 Implementasi Perangkat Lunak

Pada bagian ini akan dijelaskan implementasi proses ke dalam sebuah kode. Proses yang dibahas adalah proses-proses yang terjadi dalam sistem sesuai dengan kebutuhan fungsional.

4.2.1 Implementasi Proses Mengelola Data Praktikan

Proses ini dilakukan oleh asisten untuk mengelola data praktikan dari masing-masing kelasnya. Proses pengelolaan data praktikan dilakukan di `AccountController.php`. Implementasi proses ini menggunakan fungsi `create()` untuk menambah data praktikan, fungsi `update()` untuk mengubah data praktikan yang dipilih, `destroy()` untuk menghapus data praktikan dan fungsi `upload()` untuk mengunggah data praktikan melalui *file* Excel. Kode program fungsi-fungsi tersebut dapat dilihat pada Kode Sumber A.1 hingga Kode Sumber A.4 pada lampiran.

4.2.2 Implementasi Proses Mengelola Data *Database*

Proses ini dilakukan oleh praktikan untuk mengelola data *database* yang dapat digunakan untuk praktikum atau tutorial. Proses pengelolaan data *database* dilakukan di `DatabaseController.php`. Implementasi proses ini menggunakan fungsi `create()` untuk menambahkan data *database* serta mengunggah PDM dari *database* tersebut, fungsi `update()` untuk mengubah data *database* atau memperbarui PDM, dan fungsi `destroy()` untuk menghapus data *database*. Kode program fungsi-fungsi tersebut dapat dilihat pada Kode Sumber A.5 hingga Kode Sumber A.7 pada lampiran.

4.2.3 Implementasi Proses Mengelola Jenis *Database*

Proses ini dilakukan oleh asisten untuk mengelola jenis-jenis *database* yang dapat digunakan. Proses pengelolaan jenis

database dilakukan di `DBVersionController.php`. Implementasi proses ini menggunakan fungsi `create()` untuk menambah jenis *database* beserta parameter-parameternya dan fungsi `update()` untuk mengubah data jenis *database*. Kode program fungsi-fungsi tersebut dapat dilihat pada Kode Sumber A.8 hingga Kode Sumber A.9 pada lampiran.

4.2.4 Implementasi Proses Mengelola *Grader*

Proses ini dilakukan oleh asisten untuk mengelola *grader*. Pengelolaan *grader* dilakukan di `EventController.php`. Implementasi pengelolaan *grader* menggunakan fungsi `parserStart()` untuk memulai *grader* dan `parserStop()` untuk menghentikan *grader*. Kedua fungsi tersebut membutuhkan ID dari *grader* sebagai parameter. Kode program fungsi tersebut dapat dilihat pada Kode Sumber 4.1 hingga Kode Sumber 4.2.

```
1. $db = ListDB::with('listdbparameter')-  
  >find($id);  
2. $cmd = "python grader/grader.py ".$id." ".$db->  
  >dbversion_id." ".$db->dbversion->import." ";  
3. foreach($db->listdbparameter as $parameter) {  
4.     if($parameter->content == "") $parameter->  
      >content = "";  
5.     $cmd .= $parameter->dbversionparameter->  
      >parameter." ".$parameter->content." ";  
6. }  
7. if($process = popen("start /B ".$cmd, "r")) {  
8.     ListDB::where('id', $id)->update(array(  
9.         'status' => '1'  
10.    ));  
11.    return "true";  
12. }  
13. else {  
14.    return "false";  
15. }
```

Kode Sumber 4.1 Fungsi Memulai *Grader*

```

1.  exec("Wmic process where (Name like '%python%') get comm
    andline, ProcessID 2>&1",$output);
2.  if($output[0] == "No Instance(s) Available.")
3.  {
4.      return "false";
5.  }
6.  else
7.  {
8.      $id_arr = array();
9.      for ($i=1; $i < sizeof($output)-1; $i++) {
10.         $status = explode(" ", $output[$i]);
11.         $temp_cmd = "";
12.         for ($i=0; $i < sizeof($status); $i++) {
13.             if($status[$i]!="")
14.             {
15.                 $temp_cmd .= $status[$i]."-";
16.             }
17.         }
18.         if (strpos($status[2], 'grader')!= false)
19.         {
20.             $split = explode("-", $temp_cmd);
21.             $pid = $split[sizeof($split)-2];
22.             $command = $split[1];
23.             $grader_id = $split[2];
24.             $check = ListDB::find($grader_id);
25.             if($id == $grader_id)
26.             {
27.                 exec("Taskkill /PID ".$pid." /F");
28.             }
29.         }
30.     }
31.     ListDB::where('id', $id)->update(array(
32.         'status' => '0'
33.     ));
34.     return "true";
35. }

```

Kode Sumber 4.2 Fungsi Menghentikan Grader

4.2.5 Implementasi Proses Mengelola *Event*

Proses ini dilakukan oleh asisten untuk mengelola data praktikum dan tutorial. Selain itu, asisten juga dapat mengelola data kategori yang dapat digunakan pada tutorial. Untuk mengelola data praktikum, dilakukan pada `EventController.php` dengan fungsi `create()` untuk menambah data praktikum, fungsi `update()` untuk mengubah data praktikum dan fungsi `destroy()` untuk menghapus data praktikum. Sedangkan untuk mengelola data tutorial, dilakukan pada `EventTutorialController.php`. Untuk menambah data tutorial, digunakan fungsi `create()`, untuk mengubah data tutorial digunakan fungsi `update()` sedangkan untuk menghapus data tutorial, digunakan fungsi `destroy()`. Kemudian untuk mengelola data kategori tutorial, dilakukan pada `CategoryController.php`. Fungsi untuk menambah data kategori tutorial adalah fungsi `create()`, untuk mengubah data kategori tutorial adalah fungsi `update()` dan untuk menghapus data kategori tutorial adalah fungsi `destroy()`. Kode program fungsi-fungsi tersebut dapat dilihat pada Kode Sumber A.10 hingga Kode Sumber A.18 pada lampiran.

4.2.6 Implementasi Proses Mengelola Soal

Proses ini dilakukan asisten untuk mengelola soal yang digunakan untuk praktikum atau tutorial. Soal praktikum, dikelola di `QuestionController.php` sedangkan soal tutorial, dikelola di `TutorialController.php`. Implementasi proses pengelolaan soal praktikum menggunakan fungsi `create()` untuk menambah soal praktikum, fungsi `update()` untuk mengubah data soal dan fungsi `destroy()` untuk menghapus data soal praktikum. Sedangkan untuk soal tutorial, yang digunakan untuk menambah soal adalah fungsi `create()`, untuk mengubah data soal digunakan fungsi `update()` dan untuk menghapus data soal tutorial digunakan fungsi `destroy()`. Kode program fungsi-fungsi tersebut dapat dilihat pada Kode Sumber A.19 hingga Kode Sumber A.24 pada lampiran.

4.2.7 Implementasi Proses Melihat *Submissions*

Proses ini dilakukan oleh asisten maupun praktikan untuk melihat jawaban-jawaban yang telah terkumpul. Untuk asisten, *submissions* yang dapat dilihat adalah *submissions* dari praktikan kelasnya. Sedangkan untuk praktikan, *submissions* yang dapat dilihat adalah *submissions* miliknya sendiri. Proses untuk melihat *submissions* praktikum, dilakukan di `EventController.php` pada fungsi `viewSubmissionsSubmit()`. Sedangkan untuk melihat *submissions* tutorial, dilakukan pada `EventTutorialController.php` di fungsi `viewSubmissionsTutorialSubmit()`. Kode program fungsi tersebut dapat dilihat pada Kode Sumber 4.3 dan Kode Sumber 4.4.

```
1. $event = Event::find($id);
2. $quest_id = Question::select('id')-
   >where('event_id',$id)->get();
3. $pertanyaan = array();
4. foreach ($quest_id as $key => $value) {
5.     array_push($pertanyaan, $value->id);
6. }
7. if(Auth::user()->role_id == 3) {
8.     $submissions = Submission::whereIn('question_id', $p
   ertanyaan)->where('users_id',Auth::user()->id)->get();
9.     return view('user.event.viewSubmission', compact('su
   bmissions', 'event'));
10. }
11. else if(Auth::user()->role_id == 1 || Auth::user()-
   >role_id == 2) {
12.     $submissions = Submission::whereIn('question_id', $p
   ertanyaan)->get();
13.     return view('admin.event.viewSubmission', compact('s
   ubmissions', 'event'));
14. }
15. else {
16.     return redirect('/');
17. }
```

Kode Sumber 4.3 Fungsi Melihat *Submissions* Praktikum

```

1. $event = EventTutorial::find($id);
2. $quest_id = Tutorial::select('id')-
  >where('eventttutorial_id',$id)->get();
3. $pertanyaan = array();
4. foreach ($quest_id as $key => $value) {
5.     array_push($pertanyaan, $value->id);
6. }
7. if(Auth::user()->role_id == 3)
8. {
9.     $submissions = SubmissionTutorial::whereIn('tutorial
    _id', $pertanyaan)->where('users_id',Auth::user()->id)-
    >get();
10.    return view('user.tutorial.viewSubmission', compact(
    'submissions', 'event'));
11. }
12. else if (Auth::user()->role_id == 1 || Auth::user()-
    >role_id == 2)
13. {
14.     $submissions = SubmissionTutorial::whereIn('tutorial
    _id', $pertanyaan)->get();
15.    return view('admin.tutorial.viewSubmission', compact
    ('submissions', 'event'));
16. }
17. else
18. {
19.     return redirect('/');
20. }

```

Kode Sumber 4.4 Fungsi Melihat *Submissions* Tutorial

4.2.8 Implementasi Proses Melihat *Scoreboard*

Proses ini dilakukan oleh asisten dan praktikan untuk melihat *scoreboard* dari praktikum. Setiap pratikum mempunyai satu *scoreboard*. Proses untuk melihat *scoreboard* dilakukan pada AdminController.php yaitu fungsi scoreboardView(). Kode program fungsi tersebut dapat dilihat pada Kode Sumber 4.5

```

1. public static function cmp($a, $b) {
2.     if ($a['data']['total'] == $b['data']['total']) {
3.         return ($a['data']['nrp'] < $b['data']['nrp']) ?
4.             -1 : 1;;
5.     }
6.     return ($a['data']['total'] > $b['data']['total']) ?
7.         -1 : 1;
8. }
9.
10. public function scoreboardView($id) {
11.     $event = Event::find($id);
12.     $nilai = array();
13.     $user = User::where('kelas',$event->kelas)-
14.         >where('role_id', 3)->get();
15.     $soal = array();
16.     $question = Question::where('event_id', $id)-
17.         >orderBy('id','asc')->get();
18.     foreach ($question as $quest) {
19.         $submission = Submission::where('question_id',$q
20.             uest->id)->get();
21.         foreach ($submission as $sub) {
22.             foreach ($user as $use) {
23.                 $nilai[$use->username][$sub-
24.                     >question_id] = 0;
25.             }
26.         }
27.     }
28.
29.     foreach ($question as $quest) {
30.         foreach ($user as $use) {
31.             $submission = Submission::where('question_id
32.                 ', $quest->id)->where('users_id', $use->id)-
33.                 >orderBy('nilai', 'desc')->orderBy('created_at', 'desc')-
34.                 >first();
35.             //dd($submission->first());
36.             $nilai[$use->username][$quest-
37.                 >id] = array();
38.             if($submission)
39.             {
40.                 $nilai[$use->username][$quest-
41.                     >id]['nilai'] = $submission->nilai;
42.                 $nilai[$use->username][$quest-
43.                     >id]['created_at'] = $submission->created_at;

```

```

32.         }
33.         else $nilai[$use->username][$quest-
>id]['nilai'] = 0;
34.         $nilai[$use->username][$quest-
>id]['created_at'] = '0000-00-00 00:00:00';
35.     }
36. }
37. foreach ($nilai as $key => $value) {
38.     //dd($key);
39.     $nilai[$key]['data']['total'] = 0;
40.     $nilai[$key]['data']['nrp'] = $key;
41.     foreach ($value as $val) {
42.         if($val['nilai'] != 101) {
43.             $nilai[$key]['data']['total'] += $val['n
ilai'];
44.         }
45.     }
46. }
47. $soal_array = array_values($nilai)[0];
48. array_pop($soal_array);
49. foreach($soal_array as $quest => $value) {
50.     $pertanyaan = Question::where('id',$quest)-
>first();
51.     array_push($soal, $pertanyaan->judul);
52. }
53. $this->data['soal'] = $soal;
54. uasort($nilai, array($this, 'cmp'));
55. $this->data['nilai'] = $nilai;
56. $this->data['id'] = $id;
57. $this->data['event'] = $event;
58. if (Auth::user()->role->id == 1 or Auth::user()-
>role->id == 2) {
59.     return view('admin.scoreboard.scoreboard',$this-
>data);
60. }
61. if (Auth::user()->role->id == 3) {
62.     return view('user.scoreboard.scoreboard',$this-
>data);
63. }
64. }

```

Kode Sumber 4.5 Fungsi Melihat Scoreboard

4.2.9 Implementasi Proses Menjawab Soal *Event*

Proses ini dilakukan oleh praktikan untuk memasukkan jawaban dari soal praktikum maupun tutorial. Untuk memasukkan jawaban praktikum, dilakukan di `QuestionController.php` pada fungsi `submit()`. Sedangkan untuk menjawab soal tutorial, digunakan `TutorialController.php` pada fungsi `submit()`. Kode program fungsi tersebut dapat dilihat pada Kode Sumber 4.6 dan Kode Sumber 4.7.

```
1. $data = Input::all();
2. $jawaban = '';
3. for ($i=0;$i<=$data['jumlah'];$i++)
4. {
5.     if (isset($data['jawaban'][$i]))
6.     {
7.         $jawaban = $data['jawaban'][$i];
8.         break;
9.     }
10. }
11. Submission::create(array(
12.     'question_id' => $id2,
13.     'users_id'    => Auth::user()->id,
14.     'nilai'       => '0',
15.     'jawaban'     => $jawaban,
16.     'status'      => "0"
17. ));
```

Kode Sumber 4.6 Fungsi Menjawab Soal Praktikum

```
1. $data = Input::all();
2. $jawaban = '';
3. for ($i=0;$i<=$data['jumlah'];$i++)
4. {
5.     if (isset($data['jawaban'][$i]))
6.     {
7.         $jawaban = $data['jawaban'][$i];
8.         break;
9.     }
```



```

10. }
11. SubmissionTutorial::create(array(
12.     'tutorial_id' => $id2,
13.     'users_id'    => Auth::user()->id,
14.     'nilai'       => '0',
15.     'jawaban'     => $jawaban,
16.     'status'      => "0"
17. ));

```

Kode Sumber 4.7 Fungsi Menjawab Soal Tutorial

4.2.10 Implementasi Proses Mengelola Akun

Proses ini dilakukan praktikan untuk mengelola akun masing-masing praktikan. Pengelolaan akun terdiri dari mengubah data diri praktikan yang dilakukan pada fungsi `profile()` dan pengubahan *password* akun yang dilakukan pada fungsi `passwordResetAuth()`. Kedua fungsi tersebut terletak pada `AccountController.php`. Kode program fungsi tersebut dapat dilihat pada Kode Sumber 4.8 dan Kode Sumber 4.9

```

1. $data = Input::all();
2. User::where('id', Auth::user()->id)->update(array(
3.     'nama' => $data['nama'],
4.     'e-mail' => $data['e-mail'],
5. ));
6. Session::flash('status', 'profile-success');

```

Kode Sumber 4.8 Fungsi Mengubah Data Diri Praktikan

```

1. data = Input::all();
2. $user = User::find(Auth::user()->id);
3. if (Hash::check($data['oldPassword'], $user->password))
4. {
5.     User::where('id', $user->id)->update(array(
6.         'password' => bcrypt($data['newPassword']),
7.     ));

```

```

8.     Session::flash('status','password-success');
9.     return redirect('profile');
10. }
11. else
12. {
13.     Session::flash('status','wrong-password');
14.     return redirect('profile');
15. }

```

Kode Sumber 4.9 Fungsi Mengubah Password

4.2.11 Implementasi Proses Me-reset Password

Proses ini dilakukan oleh praktikan apabila praktikan lupa *password* dari akunnya. Apabila ingin *me-reset password*, praktikan diharuskan telah mengisikan *e-mail* pada data dirinya. Pada saat ingin *me-reset password*, praktikan harus memasukkan NRP-nya kemudian kode verifikasi yang dibuat secara *random* akan dikirimkan via *e-mail*. Proses untuk mengirimkan *e-mail* dilakukan pada fungsi `resetPassword()` dan untuk memverifikasi kode *reset password*, digunakan fungsi `verification()` pada `AccountController.php`. Kode program fungsi tersebut dapat dilihat pada Kode Sumber 4.10 dan Kode Sumber 4.11.

```

1. $data = Input::all();
2. $characters = '0123456789abcdefghijklmnopqrstuvwxyzABCDE
  FGHIJKLMNOPQRSTUVWXYZ';
3. $charactersLength = strlen($characters);
4. $randomString = '';
5. for ($i = 0; $i < 6; $i++) {
6.     $randomString .= $characters[rand(0, $charactersLeng
  th - 1)];
7. }
8. $kode = $randomString;
9. //$data['nrp'] = '5114100034';
10. $cek = User::where('username',$data['nrp'])->count();
11. if($cek > 0)
12. {

```

```

13.     $user = User::where('username',$data['nrp'])-
        >first();
14.     if($user->role_id != 3) {
15.         Session::flash('status','dont-have-right');
16.         return redirect('/');
17.     }
18.     User::where('username',$data['nrp'])-
        >update(array(
19.         'kode'      => $kode,
20.         'status'    => 1
21.     ));
22.     $title = "Password Reset ".$user->
        username." - ".$user->nama;
23.     Mail::send('resetPassword.resetPassword',[ 'nama'=> $
        title,'kode'=>$kode,'user'=>$user->
        username,'namaUser'=>$user->
        nama], function ($message) use ($title,$user,$kode){
24.         $message->to($user->e-mail)-
        >subject('Permintaan Reset Password NRP '. $user->
        username);
25.     });
26.     Session::flash('status','success');
27.     return redirect('reset/password/verification');
28. }
29. else {
30.     Session::flash('status','not-exist');
31.     return redirect('/');
32. }

```

Kode Sumber 4.10 Fungsi Mengirim Kode Verifikasi ke *E-mail*

```

1. $data = Input::all();
2. $user = User::where('username',$data['username'])-
    >first();
3. $cek = User::where('username',$data['username'])-
    >count();
4. if ($cek > 0)
5. {
6.     if ($user->role_id == 3)
7.     {
8.         if ($user->status == 1)

```

```

9.      {
10.         if($user->kode == $data['kode'])
11.         {
12.             User::where('username',$data['username'])
13.             )->update(array(
14.                 'password'      => bcrypt($data['pas
15.                 'status'       => 0,
16.             ));
17.             Session::flash('status','success');
18.             return redirect('/');
19.         }
20.         else
21.         {
22.             Session::flash('status','wrong-code');
23.             return redirect('/');
24.         }
25.         else if($user->status == 0 or $user-
26.         >status == NULL)
27.         {
28.             Session::flash('status','not-in-state');
29.             return redirect('/');
30.         }
31.         else
32.         {
33.             Session::flash('status','dont-have-right');
34.             return redirect('/');
35.         }
36.     }
37.     else if ($cek == 0)
38.     {
39.         Session::flash('status','not-exist');
40.         return redirect('/');
41.     }

```

Kode Sumber 4.11 Fungsi Verifikasi *Reset Password*

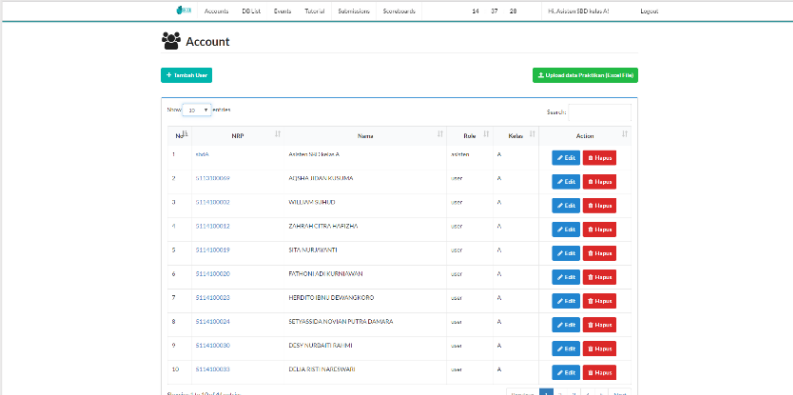
4.3 Implementasi Antarmuka Pengguna

Pada bagian ini akan dibahas implementasi antarmuka pengguna. Lapisan antarmuka merupakan lapisan yang bertugas mengatur tampilan sistem agar dapat berinteraksi dengan pengguna. Antarmuka perangkat lunak dikembangkan menggunakan beberapa pustaka. Beberapa di antaranya adalah:

- Laravel Blade sebagai skrip bahasa utama;
- Semantic UI sebagai kerangka kerja CSS;
- jQuery sebagai kerangka kerja JavaScript, dan;
- DataTables sebagai pustaka antarmuka tabel.

4.3.1 Antarmuka Mengelola Data Praktikan

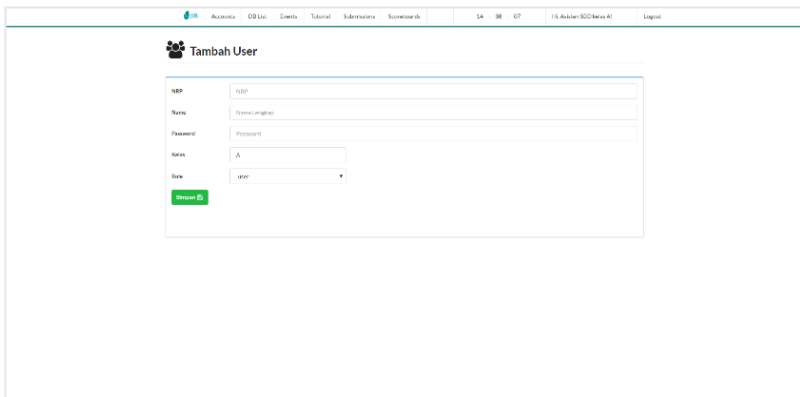
Antarmuka ini digunakan untuk kasus penggunaan mengelola data praktikan. Pada halaman mengelola data praktikan terdapat beberapa bagian, yaitu tabel untuk menampilkan data praktikan, tombol tambah data, tombol upload data menggunakan Excel, tombol ubah dan hapus data. Antarmuka untuk mengelola data praktikan dapat dilihat pada Gambar 4.1.



ID	NIP	Nama	Role	Status	Action
1	9505	ANDRIAN SUTISMAN A.	KEPERAWAT	A	Edit Delete
2	5115100009	ALYBKA IRDANA KURNIAWA	user	A	Edit Delete
3	5115100002	WILLIAM SUMEDI	user	A	Edit Delete
4	5115100052	ZAMKHA CHITRA KORTJANA	user	A	Edit Delete
5	5115100049	SITO NURJANIANTI	user	A	Edit Delete
6	5115100020	PATRICIA JODI KURNIAWATI	user	A	Edit Delete
7	5115100023	HERIPTO IBNU DEWANGKORO	user	A	Edit Delete
8	5115100004	SETYASADEN NOVIAN PUTRA DAMARA	user	A	Edit Delete
9	5115100000	DESY NUSMARTI RAUJAH	user	A	Edit Delete
10	5115100003	DELLA RISTI HANICERWATI	user	A	Edit Delete

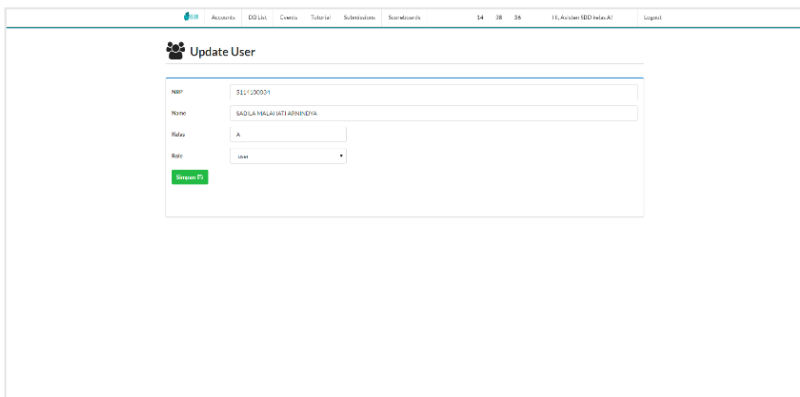
Gambar 4.1 Antarmuka Mengelola Data Praktikan

Sedangkan untuk menambah data praktikan, akan ditampilkan formulir untuk mengisi data praktikan dengan isi nama, NRP, *password* dan kelas seperti pada Gambar 4.2. Untuk mengubah data praktikan, isian formulir sama seperti saat menambah data praktikan dan ditunjukkan oleh Gambar 4.3.



The screenshot shows a web application interface with a top navigation bar containing links: Home, Accounts, DD List, Events, Tutorial, Submissions, and Scoreboards. The main content area is titled 'Tambah User' and contains a form with the following fields: NRP (text input with value '1111'), Name (text input with value 'Fenny Anggraeni'), Password (password input with value 'Password'), Kelas (dropdown menu with value 'A'), and Role (dropdown menu with value 'user'). A green 'Simpan' (Save) button is located at the bottom left of the form.

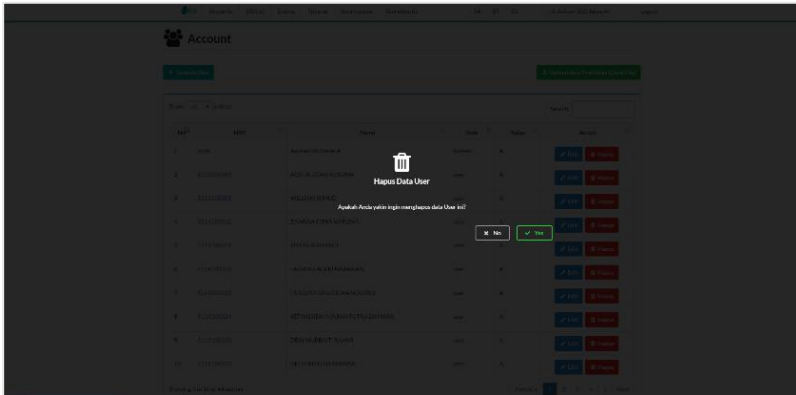
Gambar 4.2
Antarmuka Menambah Data Praktikan



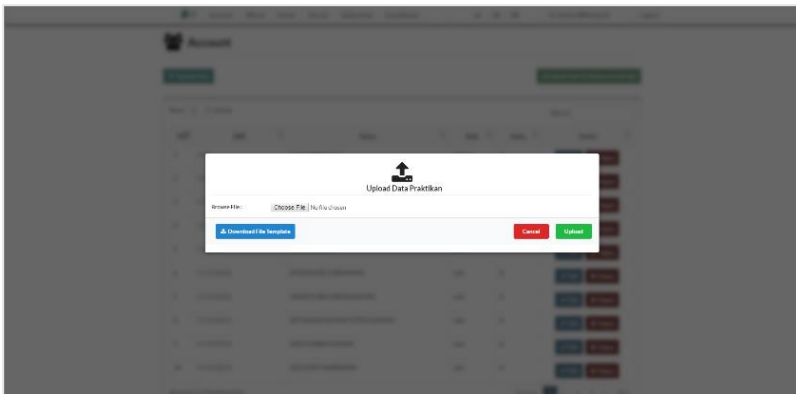
The screenshot shows a web application interface with a top navigation bar containing links: Home, Accounts, DD List, Events, Tutorial, Submissions, and Scoreboards. The main content area is titled 'Update User' and contains a form with the following fields: NRP (text input with value '5111200051'), Name (text input with value 'SADILA MALIAHATI KIRIHENDYA'), Kelas (dropdown menu with value 'A'), and Role (dropdown menu with value 'user'). A green 'Simpan' (Save) button is located at the bottom left of the form.

Gambar 4.3
Antarmuka Mengubah Data Praktikan

Untuk proses menghapus data praktikan, berupa *pop-up* konfirmasi penghapusan saat asisten menekan tombol hapus pada salah satu data praktikan. Proses ini bisa dilihat pada Gambar 4.4.



Gambar 4.4 Antarmuka Menghapus Data Praktikan



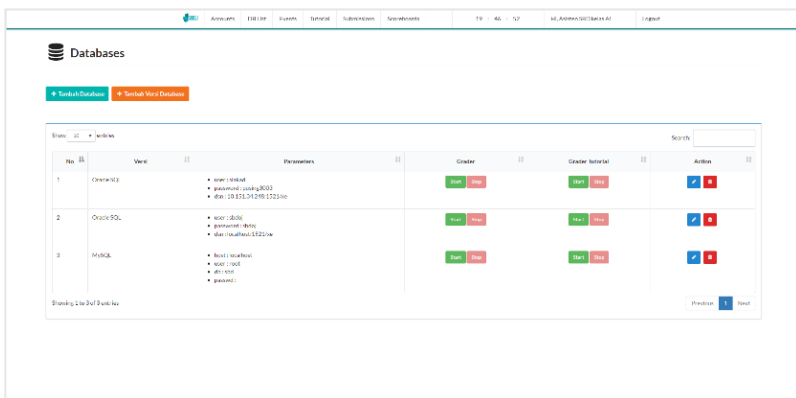
Gambar 4.5
Antarmuka *Upload File* Excel

Selain bisa menambah data melalui formulir, asisten juga bisa menambah data praktikan menggunakan *file* Excel.

Template file Excel bisa diunduh pada tombol di rancangan antarmuka “*Upload File Excel*” yang dapat dilihat pada Gambar 4.5.

4.3.2 Antarmuka Mengelola *Database*

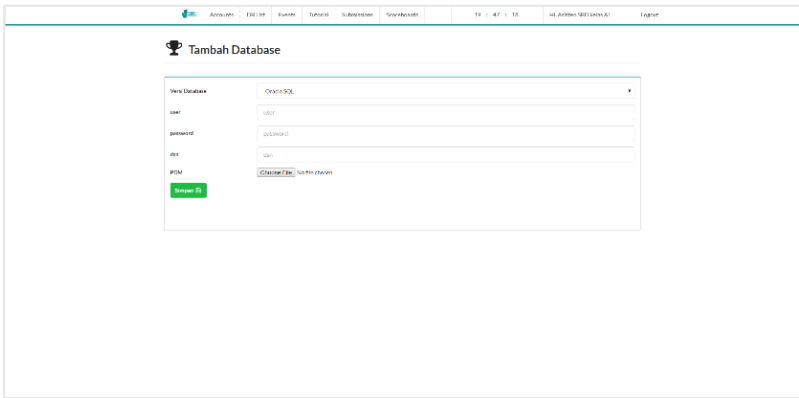
Antarmuka ini digunakan untuk kasus penggunaan mengelola *database*. Pada antarmuka tersebut terdapat halaman yang menampilkan daftar *database* yang dapat digunakan untuk praktikum maupun tutorial. Pada halaman tersebut juga digunakan untuk mengelola *database*. Pengelolaan *database* terdiri dari menambah, mengubah, menghapus *database*, menambah versi *database* dan mengelola *grader*. Antarmuka tersebut ditunjukkan oleh Gambar 4.6.



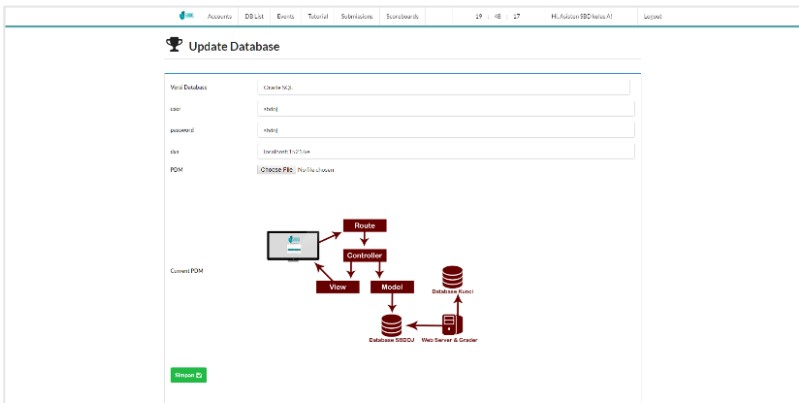
Gambar 4.6
Antarmuka Mengelola *Database*

Halaman untuk menambah *database* berisi formulir yang isinya bergantung dari parameter yang dibutuhkan oleh suatu versi *database*. Setiap *database* diharuskan mempunyai PDM. Halaman ini dapat dilihat pada Gambar 4.7. Sedangkan halaman untuk mengubah data *database* juga berupa formulir yang bergantung pada versi *database* yang dapat dilihat pada Gambar

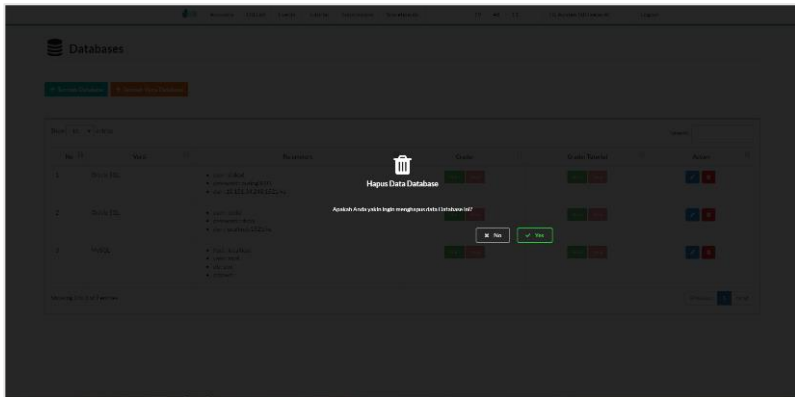
4.8. Untuk antarmuka menghapus data *database* adalah berupa *pop-up* yang digunakan untuk konfirmasi setelah asisten menekan tombol hapus. Rancangan antarmuka ditunjukkan oleh Gambar 4.9.



Gambar 4.7
Antarmuka Menambah *Database*



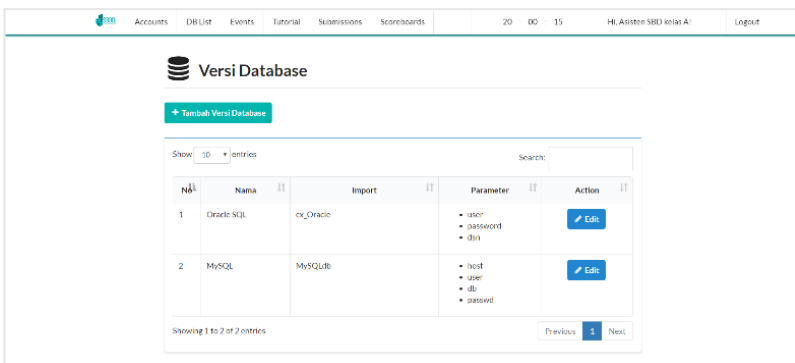
Gambar 4.8
Antarmuka Mengubah *Database*



Gambar 4.9
Antarmuka Menghapus *Database*

4.3.3 Antarmuka Mengelola Jenis *Database*

Antarmuka ini digunakan untuk kasus penggunaan mengelola jenis *database*. Pada halaman pengelolaan jenis *database* terdapat beberapa bagian, antara lain tabel untuk menampilkan daftar jenis *database*, tombol tambah dan ubah jenis *database*. Antarmuka tersebut ditunjukkan oleh Gambar 4.10.



Gambar 4.10 Antarmuka Mengelola Jenis *Database*

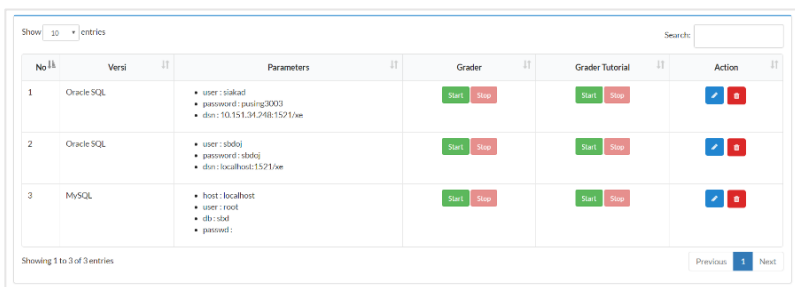
Sedangkan untuk menambah jenis *database*, terdapat form yang dinamis bergantung dari kebutuhan parameter suatu jenis *database* yang ditunjukkan oleh Gambar 4.11. Pada saat mengubah jenis *database*, asisten tidak dapat menambah parameter lagi, jadi parameter tetap seperti saat membuat jenis *database* dan dapat dilihat pada Gambar 4.12.

Gambar 4.11
Antarmuka Menambah Jenis *Database*

Gambar 4.12
Antarmuka Mengubah Jenis *Database*

4.3.4 Antarmuka Mengelola *Grader*

Antarmuka ini digunakan untuk kasus penggunaan mengelola *grader*. Pengelolaan *grader* terdiri dari memulai dan menghentikan *grader*. Setiap *database* memiliki 1 *grader* untuk praktikum dan 1 *grader* untuk tutorial. Antarmuka pengelolaan *grader* terdiri dari 2 tombol yaitu tombol Start dan Stop yang terletak di sebelah masing-masing *database*. Halaman tersebut dapat dilihat pada Gambar 4.13.

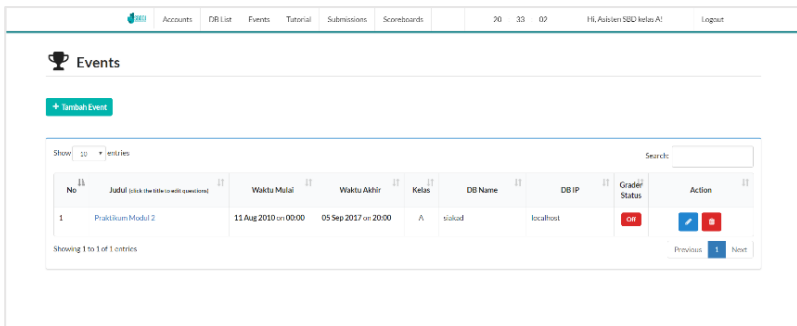


No	Versi	Parameters	Grader	Grader Tutorial	Action
1	Oracle SQL	<ul style="list-style-type: none">• user : sialad• password : puting0003• dbn : 10.151.34.248:1521/xe	Start Stop	Start Stop	✎ ✖
2	Oracle SQL	<ul style="list-style-type: none">• user : sbdol• password : sbdol• dbn : localhost:1521/xe	Start Stop	Start Stop	✎ ✖
3	MySQL	<ul style="list-style-type: none">• host : localhost• user : root• db : sbol• password :	Start Stop	Start Stop	✎ ✖

Gambar 4.13
Antarmuka Mengelola *Grader*

4.3.5 Antarmuka Mengelola Event

Antarmuka ini digunakan untuk kasus penggunaan mengelola *event*. Antarmuka pengelolaan *event* terdiri dari tabel daftar praktikum atau tutorial yang berisi judul, waktu mulai, waktu akhir, kelas, nama *database* yang dipakai, status *grader* serta tombol untuk mengubah dan menghapus *event*. Selain itu juga terdapat tombol untuk menambah *event*. Antarmuka tersebut dapat dilihat pada Gambar 4.14. Sedangkan halaman untuk menambah *event* terdiri dari sebuah formulir yang berisi judul, catatan / keterangan, tanggal mulai, waktu mulai, waktu akhir, tanggal akhir, kelas dan *database* yang digunakan. Halaman ini ditunjukkan oleh Gambar 4.15.



Gambar 4.14
Antarmuka Mengelola *Event*

Judul Event

Pesan untuk Praktikan

Catatan

Tanggal Mulai: 18/04/2017

Waktu Mulai: 20:33:00

Waktu Akhir: 20:32:00

Tanggal Akhir: 18/04/2017

Kelas: A

Database: 1

Simpan

Gambar 4.15
Antarmuka Menambah *Event*

Sedangkan untuk mengubah data *event*, terdapat halaman yang berisi formulir yang sama dengan formulir saat membuat *event* dan dapat dilihat pada Gambar 4.16. Untuk menghapus data *event*, digunakan *pop-up* untuk konfirmasi penghapusan seperti pada Gambar 4.17. Untuk halaman pengelolaan kategori tutorial, dapat dilihat pada Gambar 4.18. Pengelolaan kategori berupa tabel yang berisi nama kategori dan tombol ubah serta

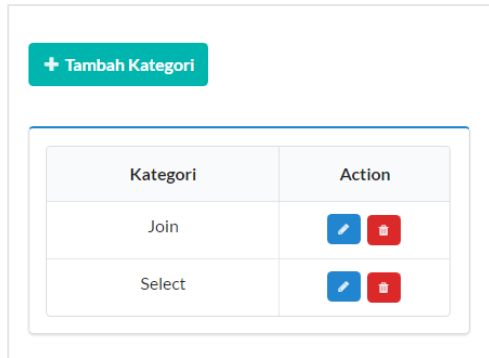
hapus data kategori. Untuk menambah kategori, digunakan formulir yang berisi nama kategori dan penjelasan dari kategori tersebut yang ditunjukkan oleh Gambar 4.19.

The screenshot shows a web application interface with a top navigation bar containing links: Accounts, User List, Events, Tutorial, Submissions, and Scoreboards. On the right of the navigation bar, there are user details: '20 34 00', 'Hi, Asisten S&D Hesti A!', and a 'Logout' button. The main content area is titled 'Update Event' with a trophy icon. It contains a form with the following fields: 'Judul Event' (Praktikum Modul 2), 'Catatan' (a list of instructions for users), 'Tanggal Mulai' (11-08-2018), 'Waktu Mulai' (00:00:00), 'Waktu Akhir' (20:00:00), 'Tanggal Akhir' (03-09-2017), 'Kelas' (A), and 'Database' (stake | location). A green 'Simpan ID' button is at the bottom left of the form.

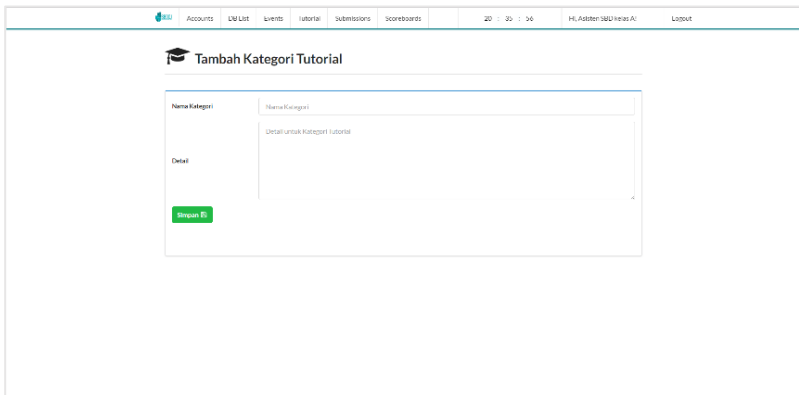
Gambar 4.16
Antarmuka Mengubah *Event*

The screenshot shows a dark-themed web application interface. At the top, there is a navigation bar with links: Accounts, User List, Events, Tutorial, Submissions, and Scoreboards. On the right, there are user details: '20 34 13', 'Hi, Asisten S&D Hesti A!', and a 'Logout' button. The main content area is titled 'Events' with a trophy icon. Below the title, there is a 'Hapus Data Event' dialog box. The dialog box has a title bar with a trash icon and the text 'Hapus Data Event'. It contains a message: 'Apakah Anda yakin ingin menghapus data Event ini?'. Below the message, there are two buttons: 'No' and 'Yes'. The 'Yes' button is highlighted with a green checkmark.

Gambar 4.17
Antarmuka Menghapus *Event*



Gambar 4.18 Antarmuka Mengelola Kategori Tutorial

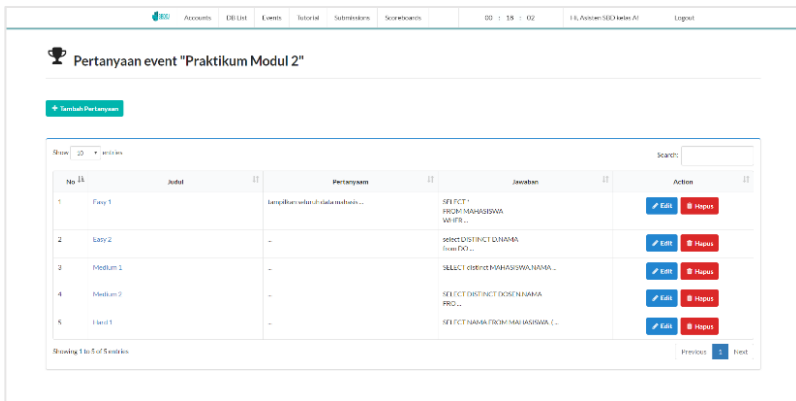


Gambar 4.19 Antarmuka Menambah Kategori Tutorial

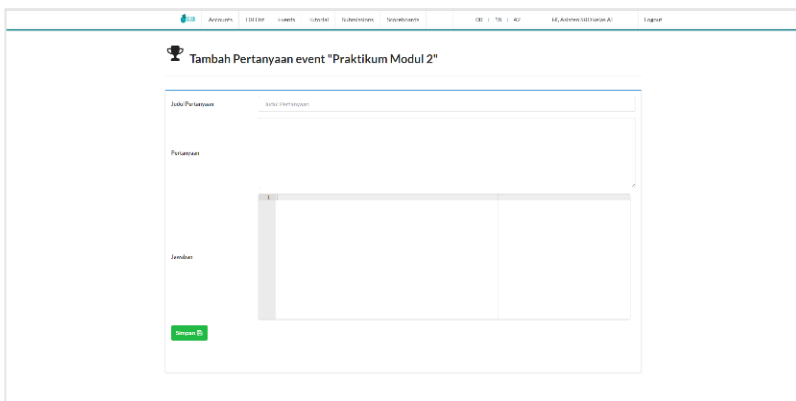
4.3.6 Antarmuka Mengelola Soal

Antarmuka ini digunakan untuk kasus penggunaan mengelola soal praktikum atau tutorial. Pada antarmuka tersebut terdapat beberapa halaman. Halaman mengelola soal terdapat beberapa bagian, yaitu tombol untuk menambah soal, daftar soal yang ditampilkan dalam bentuk tabel, dan tombol untuk menghapus dan mengubah data soal. Antarmuka untuk mengelola soal dapat dilihat pada Gambar 4.20. Halaman tambah

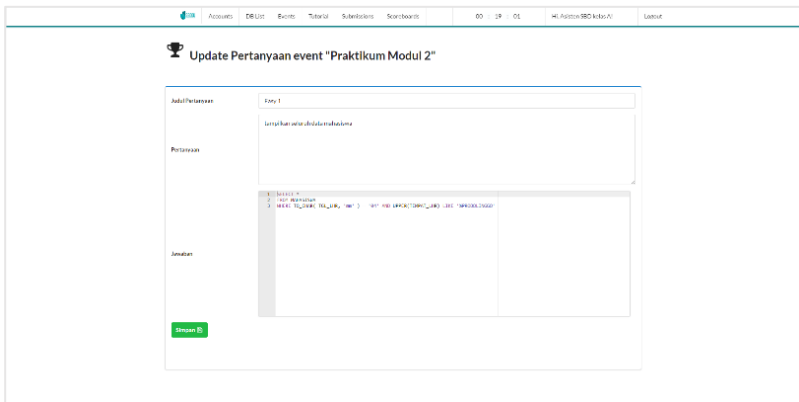
data soal menampilkan formulir yang berisi judul soal, pertanyaan, dan jawaban dari soal yang dapat dilihat pada Gambar 4.21. Untuk halaman mengubah data soal, menampilkan formulir untuk mengubah data soal dengan isi sama seperti formulir tambah soal dan dapat dilihat pada Gambar 4.22. Sedangkan antarmuka untuk menghapus data soal berupa *pop-up* seperti pada Gambar 4.23.



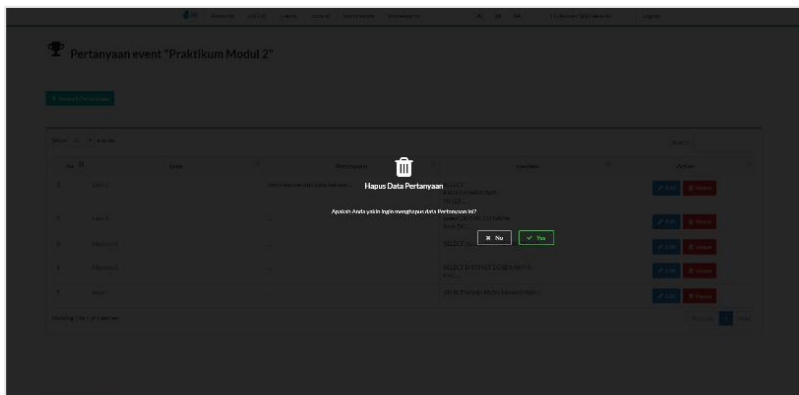
Gambar 4.20 Antarmuka Mengelola Soal



Gambar 4.21 Antarmuka Tambah Soal



Gambar 4.22
Antarmuka Ubah Soal



Gambar 4.23
Antarmuka Hapus Soal

4.3.7 Antarmuka Melihat *Submissions*

Antarmuka ini digunakan untuk kasus penggunaan melihat *submissions*. Halaman *submissions* menampilkan tabel dengan isi daftar *submissions* dari praktikan dalam suatu praktikum atau

tutorial. Terdapat pembeda pada tabel tersebut, yaitu apabila jawaban salah maka akan diberi latar warna merah sedangkan untuk jawaban benar diberi warna hijau. Rancangan antarmuka *submissions* dapat dilihat pada Gambar 4.24.

[illegible]

Gambar 4.24
Antarmuka Daftar *Submissions*

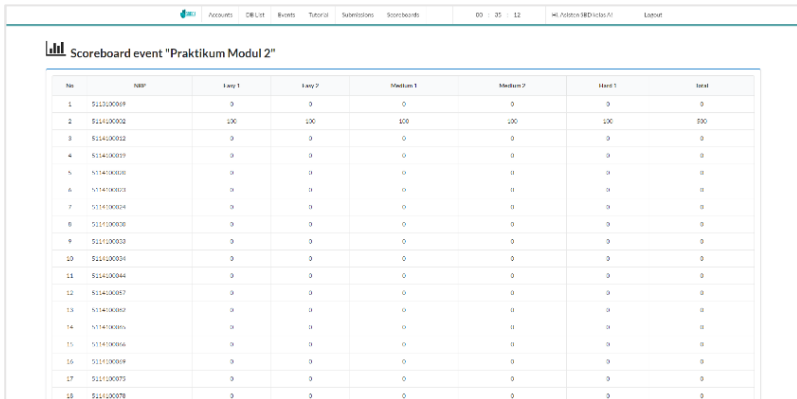
4.3.8 Rancangan Antarmuka Melihat *Scoreboard*

Antarmuka ini digunakan untuk kasus penggunaan melihat *scoreboard*. Halaman ini menampilkan tabel *scoreboard* dari suatu praktikum. Pada tabel tersebut terdapat nilai dari masing-masing praktikan pada masing-masing soal dan juga terdapat total nilai dari tiap praktikan. Antarmuka *scoreboard* dapat dilihat pada Gambar 4.25.

4.3.9 Antarmuka Menjawab Soal *Event*

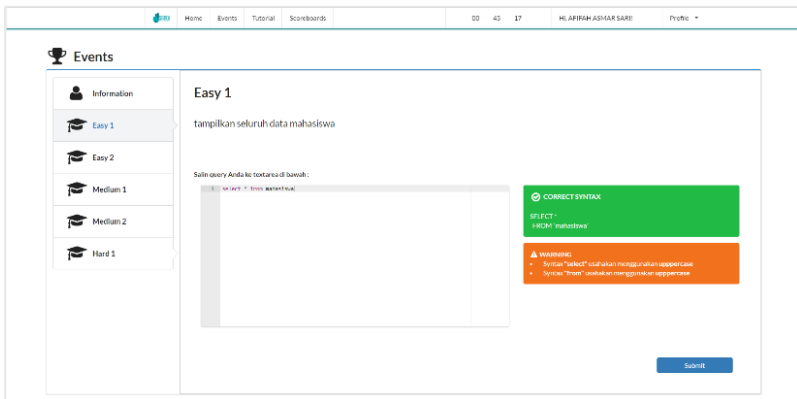
Antarmuka ini digunakan untuk kasus penggunaan menjawab soal praktikum dan tutorial. Pada antarmuka tersebut soal ditampilkan dalam bentuk daftar tabulasi dan pada masing-masing soal terdapat *field* untuk memasukkan jawaban praktikan. Pada masing-masing soal juga terdapat SQL Parser untuk

mengecek benar atau tidaknya *syntax* SQL dari praktikan. Antarmuka menjawab soal *event* dapat dilihat pada Gambar 4.26.



No	NIM	Easy 1	Easy 2	Medium 1	Medium 2	Hard 1	Total
1	5111200019	0	0	0	0	0	0
2	5111200002	100	100	100	100	100	500
3	5111200032	0	0	0	0	0	0
4	5111200037	0	0	0	0	0	0
5	5111200001	0	0	0	0	0	0
6	5111200003	0	0	0	0	0	0
7	5111200004	0	0	0	0	0	0
8	5111200008	0	0	0	0	0	0
9	5111200003	0	0	0	0	0	0
10	5111200004	0	0	0	0	0	0
11	5111200044	0	0	0	0	0	0
12	5111200087	0	0	0	0	0	0
13	5111200067	0	0	0	0	0	0
14	5111200006	0	0	0	0	0	0
15	5111200006	0	0	0	0	0	0
16	5111200019	0	0	0	0	0	0
17	5111200073	0	0	0	0	0	0
18	5111200078	0	0	0	0	0	0

Gambar 4.25 Antarmuka Scoreboard



Events

Easy 1

tampilkan seluruh data mahasiswa

Solusi query Anda ke kotak ini di bawah:

```
select * from mahasiswa
```

CORRECT SYNTAX

Warning

- Syntax 'select*' is not an accepted uppercase
- Syntax 'from' is not an accepted uppercase

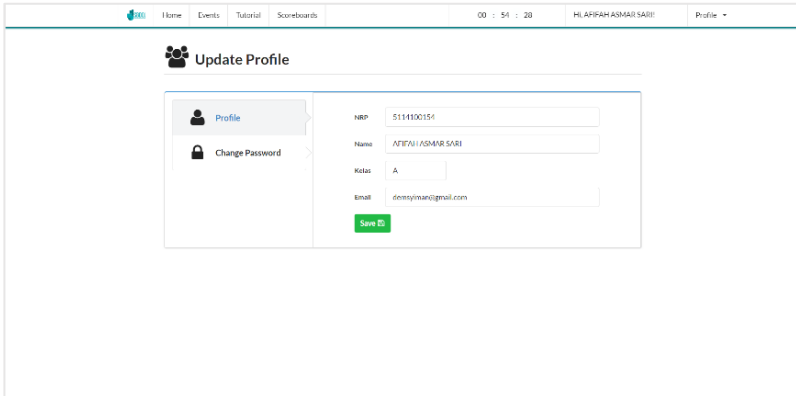
Submit

Gambar 4.26
Antarmuka Menjawab Soal Event

4.3.10 Antarmuka Mengelola Akun

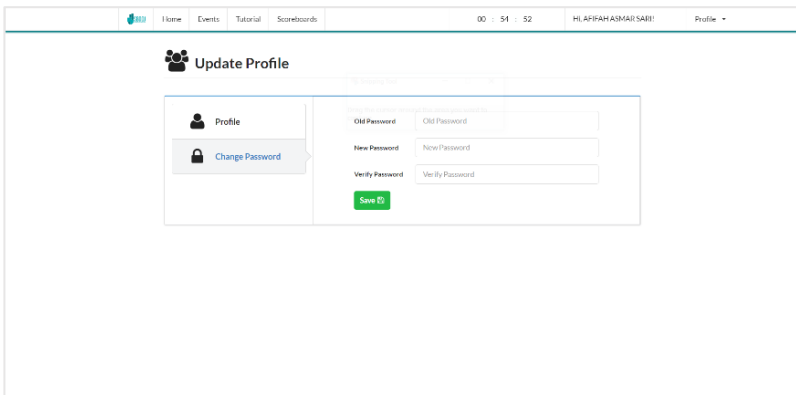
Antarmuka ini digunakan untuk kasus penggunaan mengelola akun praktikan. Pada antarmuka tersebut terdapat

beberapa menu. Menu pertama adalah untuk mengubah data diri praktikan. Menu ini menampilkan formulir dengan isi NRP, nama, kelas dan juga *e-mail* dari praktikan yang dapat dilihat pada Gambar 4.27.



The screenshot shows a web application interface for updating a user's profile. At the top, there is a navigation bar with links for Home, Events, Tutorial, and Scoreboards. The user's name, HI AFIRAH ASMAR SARI, and a profile dropdown menu are visible on the right. The main content area is titled 'Update Profile' and contains a sidebar with two options: 'Profile' (selected) and 'Change Password'. The 'Profile' section displays a form with the following fields: NRP (5111100154), Name (AFIFAI ASMAR SARI), Kelas (A), and Email (denyswan@gmail.com). A green 'Save' button is located at the bottom right of the form.

Gambar 4.27 Antarmuka Mengubah Data Diri Praktikan



The screenshot shows the same web application interface, but with the 'Change Password' section selected in the sidebar. The 'Profile' section is now inactive. The 'Change Password' section displays a form with three fields: Old Password, New Password, and Verify Password. A green 'Save' button is located at the bottom right of the form.

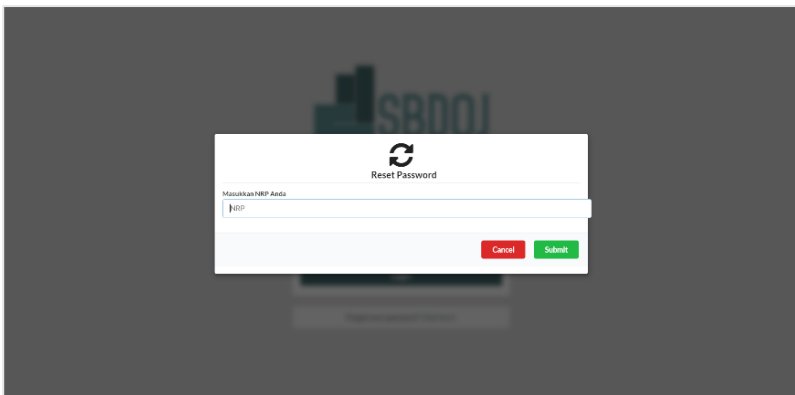
Gambar 4.28 Antarmuka Mengubah *Password*

Sedangkan menu selanjutnya adalah mengubah *password*. Praktikan dapat mengubah *password*-nya dengan mengisi formulir pada halaman ini. Formulir tersebut berisi *password*

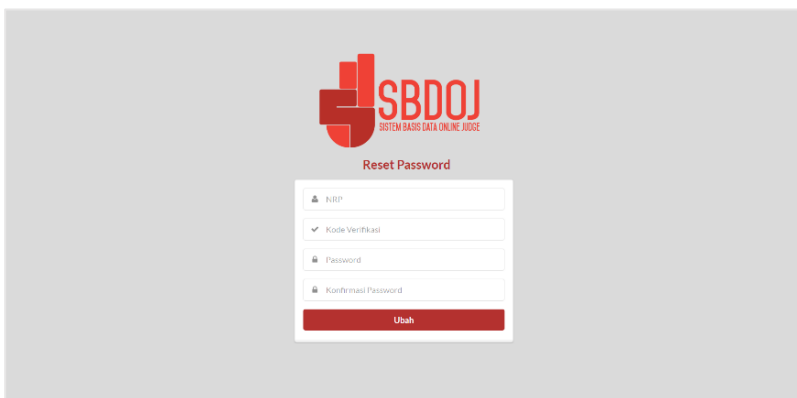
lama, *password* baru dan konfirmasi *password* baru. Menu ini dapat dilihat pada Gambar 4.28.

4.3.11 Antarmuka Me-reset *Password*

Antarmuka ini digunakan untuk kasus penggunaan *me-reset password*. Pada antarmuka tersebut terdapat *pop-up* untuk praktikan memasukkan NRP yang dapat dilihat pada Gambar 4.29.



Gambar 4.29 Antarmuka Memasukkan NRP



Gambar 4.30 Antarmuka *Reset Password*

Setelah praktikan mengirim NRP, maka akan diarahkan ke halaman *reset password* yang menampilkan formulir dengan isi NRP, kode verifikasi, *password* baru dan konfirmasi *password* baru. Antarmuka tersebut dapat dilihat pada Gambar 4.30.

.

BAB V

PENGUJIAN DAN EVALUASI

Pada bab ini dijelaskan pengujian dan evaluasi dari aplikasi SBDOJ. Pengujian yang dilakukan adalah pengujian *user acceptance test* dan pengujian system. Pengujian sistem difokuskan pada *grader* atau proses *judgement*. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini.

5.1 Lingkungan Pelaksanaan Pengujian

Lingkungan pengujian aplikasi SBDOJ dilakukan dengan spesifikasi sebagai berikut:

Jenis : Personal Computer
Tipe : LENOVO 10132
Prosesor : Intel® Core™ i3-4150 CPU @ 3.50GHz
Memori : 4096MB RAM
Sistem Operasi : Windows 10 Pro 64-bit

5.2 Pengujian *User Acceptance Test*

Pengujian *User Acceptance Test* yang pada aplikasi SBDOJ bertujuan untuk mengetahui tingkat kebutuhan asisten akan aplikasi SBDOJ dan tingkat kesesuaian SBDOJ dengan kebutuhan praktikum Sistem Basis Data khususnya materi DML *query select* di Departemen Teknik Informatika ITS. Jumlah responden asisten adalah 5 orang dengan 7 pertanyaan.

Kategori penilaian yang digunakan untuk kuesioner dapat dilihat pada Tabel 5.1.

Tabel 5.1 Kategori Penilaian UAT

Jawaban	Keterangan	Bobot
STS	Sangat Tidak Setuju	1

TS	Tidak Setuju	2
N	Netral	3
S	Setuju	4
SS	Sangat Setuju	5

Sedangkan kuesioner untuk pengujian *User Acceptance Test* dapat dilihat pada Tabel 5.2.

Tabel 5.2 Kuesioner Pengujian UAT

No	Pertanyaan	STS	TS	N	S	SS
1	SBDOJ dapat dioperasikan dengan mudah oleh asisten					
2	SBDOJ sudah memenuhi kebutuhan praktikum SBD materi DML <i>query select</i>					
3	SBDOJ memudahkan asisten dalam praktikum SBD materi DML <i>query select</i>					
4	SBDOJ dapat menghemat waktu praktikum SBD					
5	Proses penilaian SBDOJ sudah benar					
6	Asisten membutuhkan SBDOJ untuk praktikum SBD materi DML <i>query select</i>					
7	SBDOJ hendaknya digunakan seterusnya pada saat praktikum SBD materi DML <i>query select</i>					

Setelah kuesioner diatas diberikan kepada asisten, kemudian data kuesioner tersebut diolah untuk mendapatkan hasil penilaian *user acceptance test*. Untuk data kuesioner *user acceptance test* dapat dilihat pada Lampiran B.1 sampai Lampiran B.5 . Adapun hasil penilaian *user acceptance test* tersebut yaitu sebagai berikut :

Tabel 5.3 Hasil Perhitungan UAT

Pertanyaan	Jawaban Asisten									
	STS	%	TS	%	N	%	S	%	SS	%
1	0	0%	0	0%	0	0%	4	80%	1	20%
2	0	0%	0	0%	0	0%	4	80%	1	20%
3	0	0%	0	0%	0	0%	3	60%	2	40%
4	0	0%	0	0%	0	0%	2	40%	3	60%
5	0	0%	0	0%	1	20%	2	40%	2	40%
6	0	0%	0	0%	0	0%	2	40%	3	60%
7	0	0%	0	0%	0	0%	0	0%	5	100%
Total	0	0	0	0	1	2.86%	17	48.6%	17	48.6%

5.3 Pengujian Sistem

Pada bagian ini akan dijelaskan tentang skenario pengujian yang dilakukan pada SBDOJ. Pengujian yang dilakukan adalah pengujian keberhasilan proses judgement dan kemampuan menangani *query* yang memungkinkan terjadinya eror pada sistem. Pengujian didasarkan pada dua versi *grader*. Pengujian dilakukan dengan data-data yang telah dikumpulkan saat praktikum Sistem Basis Data dari tahun ajaran 2015/2016 dan 2016/2017.

5.3.1 Pengujian Tingkat Keberhasilan *Grader*

Pengujian tingkat keberhasilan *grader* bertujuan untuk mengetahui hasil keluaran dari *grader*, apakah sesuai atau tidak dengan yang diharapkan. Terdapat 2 versi *grader* yang telah dikembangkan berdasarkan kasus-kasus yang dialami selama pengujian. Pengujian ini dilakukan dengan mencoba berbagai macam *query* yang memungkinkan eror pada saat proses *judgement*. Jenis-jenis *query* yang akan diuji adalah:

1. *Query DML insert*. Contoh :

```
INSERT INTO mahasiswa VALUES('5113100015','Densy Iman M','L',
'20-FEB-95','Kabupaten Lumajang',1)
```

2. *Query DML update.* Contoh :

```
UPDATE mahasiswa SET nama='Mahasiswa Coba Update'
WHERE nrp = '5104100122'
```

3. *Query DML delete.* Contoh :

```
DELETE FROM mahasiswa where nrp='5104100122'
```

4. *Complex Query.* Contoh :

```
SELECT DISTINCT f.id_mk_tawar, f.nrp, f.nilai_huruf FROM fr
s f, mahasiswa m, mk_tawar mkt WHERE mkt.id_mk='IF1412' AND
(SELECT n.nilai_angka FROM nilai n WHERE f.nilai_huruf=n.n
ilai_huruf)>(SELECT n.nilai_angka FROM nilai n WHERE nilai_
huruf='B') AND m.nrp=f.nrp AND mkt.id_mk_tawar=f.id_mk_tawa
r AND m.jk='L' ORDER BY f.id_mk_tawar
```

5. *Query dengan hasil yang sangat banyak (> 5.000.000.000 rows).* Contoh :

```
SELECT COUNT(*) FROM (SELECT MT.* FROM matakuliah MT, frs F
, mahasiswa M WHERE EXTRACT(MONTH FROM m.tgl_lhr)=09 ORDER
BY f.id_mk_tawar ASC)
```

Jenis-jenis *query* tersebut ditentukan berdasarkan jenis-jenis error yang mungkin menyebabkan *grader* menjadi error atau menghambat eksekusi *grader*.

5.3.1.1 Pengujian Grader Versi I

Grader versi I adalah *grader* yang digunakan pada saat praktikum Sistem Basis Data di Teknik Informatika ITS tahun ajaran 2015/2016. *Grader* versi I dapat dilihat pada Kode Sumber A.25. Hasil pengujian dapat dilihat pada Tabel 5.4.

Tabel 5.4 Hasil Pengujian Grader Versi I

Jenis <i>Query</i>	Output	Keterangan
<i>Query DML insert</i>	<i>Exception</i>	Sukses
<i>Query DML update</i>	<i>Exception</i>	Sukses

<i>Query DML delete</i>	<i>Exception</i>	Sukses
<i>Complex query</i>	<i>Success</i>	Sukses
<i>Query dengan hasil null</i>	<i>Success</i>	Sukses
Query dengan hasil > 5.000.000.000 row	-	Membutuhkan waktu yang sangat lama untuk proses <i>judgment</i>

Pada tabel di atas, dapat dilihat bahwa *query insert*, *update*, dan *delete* masuk ke dalam *Exception*, karena memang SBDOJ hanya untuk *query DML select*, sehingga tidak bisa mengeksekusi *query* yang dapat mengubah data pada *database*. Kemudian untuk *complex query*, *grader* versi I dapat mengeksekusinya dengan lancar. Sedangkan untuk *query* yang menghasilkan banyak sekali *row*, *grader* ini membutuhkan waktu yang sangat lama untuk proses *judgment*.

Kekurangan dari *grader* versi I adalah :

1. Tidak dapat mengeksekusi *query* apabila *query* tersebut mengandung *semicolon*.
2. Belum ada penanganan untuk *query* yang menghasilkan *row* sangat banyak, sehingga menghambat eksekusi *query* selanjutnya.
3. Karena belum adanya penanganan untuk kasus nomor 2, pada saat pengujian, server menjadi *down* sebanyak 2 kali.
4. Satu *grader* hanya untuk satu jenis *database*.
5. *Grader* mengecek dari ID *submission* yang paling rendah, tidak berdasarkan ID dari *database* yang sedang digunakan.

5.3.1.2 Pengujian Grader Versi II

1. *Grader* versi II adalah *grader* yang digunakan pada saat praktikum Sistem Basis Data di Teknik Informatika ITS

tahun ajaran 2016/2017. *Grader* versi II dapat dilihat pada `#!/usr/bin/python\`

```
2. import sys
3. import os
4. from multiprocessing import Process
5. import time
6. import threading
7. import MySQLdb
8.
9. moduleNames = [str(sys.argv[3])]
10. modules = map(__import__, moduleNames)
11. dct = {}
12. for x in xrange(4, len(sys.argv), 2):
13.     if (sys.argv[x+1] == ""): sys.argv[x+1] = ""
14.     dct[str(sys.argv[x])] = sys.argv[x+1]
15. db_kunci = modules[0].connect(**dct)
16. cursor_kunci = db_kunci.cursor()
17.
18. def check(submission_id):
19.     try:
20.         db = MySQLdb.connect('localhost', 'root', '
', 'sbd')
21.         cursor = db.cursor()
22.         sql = '''select question_id, jawaban from
submission where id = ''' + str(submission_id)
23.         cursor.execute(sql)
24.         unchecked = cursor.fetchone()
25.         sub_id = submission_id
26.         ques_id = ''
27.         ans = ''
28.         tanda = 0
29.
30.         while unchecked is not None:
31.             ques_id = unchecked[0]
32.             ans = unchecked[1]
33.             unchecked = cursor.fetchone()
34.             ans = ans.replace('; ', '')
35.             cursor_kunci.execute(ans)
36.             results = cursor_kunci.fetchall()
37.             num_fields = len(cursor_kunci.description)
```

```

38.         hasil=[[0 for x in range(num_fields)] for
x in range(10000)]
39.         rows=0
40.         for res in results:
41.             for column in range(num_fields):
42.                 hasil[rows][column] = res[column]

43.             rows+=1
44.
45.
46.         kunci = '''select jawaban from question wh
ere id='''+ str(ques_id)
47.         cursor.execute(kunci)
48.         temp = cursor.fetchone()
49.         while temp is not None:
50.             temp_kunci = temp[0]
51.             temp_kunci = temp_kunci.replace(';',' '
')
52.             temp = cursor.fetchone()
53.             cursor_kunci.execute(temp_kunci)
54.             res_kunci = cursor_kunci.fetchall()
55.             num_fields_1 = len(cursor_kunci.descriptio
n)
56.             arr_kunci=[[0 for x in range(num_fields_1)
] for x in range(10000)]
57.             row_kunci=0
58.             for res_key in res_kunci:
59.                 for column_kunci in range(num_fields_1
):
60.                     arr_kunci[row_kunci][column_kunci]
= res_key[column_kunci]
61.                     row_kunci+=1
62.
63.
64.             flag=0
65.             if (num_fields != num_fields_1):
66.                 flag=1
67.             if (rows != row_kunci):
68.                 flag=1
69.             if (flag==0):
70.                 for row_compare in range(row_kunci):
71.                     for column_compare in range(num_fi
elds):

```

```

72.                 if (hasil[row_compare][column_
compare] != arr_kunci[row_compare][column_compare]):
73.                     flag=1
74.                 if (flag==1):
75.                     update1 = '''update submission set nil
ai = 0, status = 1 where id = ''' + str(sub_id)
76.                     cursor.execute(update1)
77.                     db.commit()
78.                     print 'query ' + str(sub_id) + ' failed'
79.
80.                 elif (flag==0):
81.                     update2 = '''update submission set nil
ai = 100, status = 1 where id = ''' + str(sub_id)
82.                     cursor.execute(update2)
83.                     db.commit()
84.                     print 'query ' + str(sub_id) + ' success'
85.                 return
86.
87.             except :
88.                 #print str(sub_id) + ' Failed : ' + str(e) +
' ' + str(temp_kunci)
89.                 print 'Exception ' + str(sub_id)
90.                 db.rollback()
91.                 update1 = '''update submission set nilai =
0, status = 1 where id = ''' + str(sub_id)
92.                 cursor.execute(update1)
93.                 db.commit()
94.                 return
95.
96. def stopwatch(timeout, target, args):
97.     p = Process(target=target, args=args)
98.     sub_id = args[0]
99.     p.start()
100.    for x in xrange(0, timeout):
101.        time.sleep(1)
102.        #print p.is_alive()
103.        if (p.is_alive() == False):
104.            return
105.        print 'TLE ' + str(sub_id)
106.        db.rollback()

```

```

107.         updateTLE = '''update submission set n
            ilai = 101, status = 1 where id = ''' + str(sub_id)

108.         cursor.execute(updateTLE)
109.         db.commit()
110.         p.terminate()
111.
112.         if __name__ == '__main__':
113.             try:
114.                 while True:
115.                     db= MySQLdb.connect('localhost
116. ', 'root', '', 'sbd')
117.                     cursor = db.cursor()
118.                     #try:
119.                     sql = '''select s.id, s.questi
120. on_id, s.users_id, s.jawaban, s.status from submis
121. sion s,event e, listdb ldb, question q where s.que
122. stion_id = q.id and q.event_id = e.id and e.listdb
123. _id = ldb.id and s.status = 0 and ldb.dbversion_id
124. = ''' + sys.argv[2] + ''' having s.id = min(s.id)'''
125.
126.                     cursor.execute(sql)
127.                     unchecked = cursor.fetchone()
128.
129.                     sub_id = ''
130.                     ques_id = ''
131.                     user_id = ''
132.                     ans = ''
133.                     stat = ''
134.                     tanda = 0
135.                     while unchecked is not None:
136.                         sub_id = unchecked[0]
137.                         ques_id = unchecked[1]
138.                         user_id = unchecked[2]
139.                         ans = unchecked[3]
140.                         stat = unchecked[4]
141.                         unchecked = cursor.fetchon
142. e()
143.                     if (sub_id!='') :
144.                         tanda = 1
145.                     if (tanda==1):
146.                         t = threading.Thread(targe
147. t=stopwatch, args=(20,check,(str(sub_id),)))

```

```

138.                 t.start()
139.                 time.sleep(1)
140.                 t.join()
141.                 print "next"
142.                 db.close()
143.                 db_kunci.close()
144.
145.             except KeyboardInterrupt:
146.                 sys.exit(0)
147.         except :
148.             print 'berhenti'

```

Kode Sumber A.26. Hasil pengujian dapat dilihat pada Tabel 5.5.

Tabel 5.5 Hasil Pengujian Grader Versi II

Jenis Query	Output	Keterangan
<i>Query DML insert</i>	<i>Exception</i>	Sukses
<i>Query DML update</i>	<i>Exception</i>	Sukses
<i>Query DML delete</i>	<i>Exception</i>	Sukses
<i>Complex query</i>	<i>Success</i>	Sukses
<i>Query dengan hasil null</i>	<i>Success</i>	Sukses
Query dengan hasil > 5.000.000.000 row	TLE	Sukses

Untuk *query insert*, *update*, *delete* dan *complex query*, hasil keluarannya sudah sesuai. Kemudian untuk *query* yang hasilnya memiliki *row* sangat banyak, sudah dapat mengeluarkan hasil *time limit exceeded* (TLE) setelah 10 detik. Perhitungan waktu tersebut dilakukan dengan menggunakan *thread*.

Kelebihan *grader* versi II dibandingkan *grader* versi I adalah :

1. Sudah dapat mengeksekusi *query* yang mengandung *semicolon*, karena sudah ditambahkan kode untuk menghilangkan *semicolon* sebelum *query* tersebut dieksekusi.
2. Sudah dapat menangani *query* dengan hasil yang sangat banyak dengan menambahkan *timer* dengan waktu eksekusi maksimal adalah 10 detik. Setelah melewati 10 detik, maka *query* tersebut dianggap TLE.
3. Untuk pengujian kedua ini, server sudah tidak mengalami *down* Karena sudah ada penanganan TLE.
4. Satu *grader* bisa digunakan untuk berbagai jenis *database* Karena parameter-parameter tiap jenis *database* disimpan dalam *database* sehingga parameter-parameter tersebut dikirim saat pemanggilan *grader*.
5. Pemanggilan *grader* disertai dengan pengiriman ID *database*, sehingga *grader* tidak perlu mengecek dari ID *submission* yang paling rendah, hanya mengecek *submissions* yang soal-soalnya menggunakan *database* sesuai ID yang dikirimkan.

5.4 Evaluasi Pengujian

Pada bagian ini akan dijelaskan hasil evaluasi dari pengujian-pengujian yang telah dilakukan sebelumnya. Evaluasi yang diberikan meliputi evaluasi pengujian tingkat keberhasilan *grader* dalam menilai jawaban-jawaban dari praktikan.

5.4.1 Evaluasi Pengujian *User Acceptance Test*

Berdasarkan hasil pengujian *user acceptance test*, dapat dilihat bahwa pengoperasian SBDOJ sudah mudah untuk digunakan, sudah memenuhi kebutuhan praktikum SBD materi DML *query select*, dapat memudahkan dan menghemat waktu asisten. Responden juga membutuhkan SBDOJ pada praktikum SBD materi DML *query select*. Namun terdapat usulan dari responden, untuk proses penilaian SBDOJ seharusnya

ditambahkan bobot nilai tiap soal, sehingga asisten dapat memberikan nilai yang spesifik berdasar kemampuan praktikan.

5.4.2 Evaluasi Pengujian Tingkat Keberhasilan

Berdasarkan pada pengujian *grader* versi I dan *grader* versi II, dapat dilihat bahwa kekurangan-kekurangan yang dialami oleh *grader* versi I sudah dapat ditangani oleh *grader* versi II. Namun, masih terdapat 1 kekurangan yang dialami oleh *grader* versi II, yaitu waktu maksimal eksekusi masih statis, 10 detik. Setelah 10 detik maka *query* tersebut dianggap TLE. Seharusnya waktu maksimal eksekusi dibuat dinamis berdasarkan soal atau jumlah *row* hasil *query* yang dibutuhkan.

[Halaman sengaja dikosongkan]

BAB VI

KESIMPULAN DAN SARAN

Pada bab ini diberikan kesimpulan yang diambil selama pengerjaan tugas akhir beserta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1 Kesimpulan

Selama proses perancangan, implementasi, dan pengujian dapat diambil kesimpulan sebagai berikut:

1. Aplikasi SBDOJ mampu menangani prosedur penilaian *query* jawaban dari praktikan dan dicocokkan dengan *query* kunci jawaban dari asisten.
2. Proses bisnis yang terjadi antara asisten dan praktikan dalam prosedur penilaian jawaban praktikan adalah sebagai berikut:
 - a. Asisten memasukkan data *database* yang akan digunakan untuk praktikum atau tutorial.
 - b. Asisten memasukkan data praktikum atau tutorial.
 - c. Asisten memasukkan data soal praktikum atau tutorial.
 - d. Asisten menekan tombol *start* pada *grader* yang dibutuhkan.
 - e. Praktikan memilih praktikum atau *tutorial* yang diinginkan.
 - f. Praktikan memilih soal yang diinginkan.
 - g. Praktikan memasukkan *query* jawabannya.
 - h. *Grader* mengecek dan menilai *query* jawaban praktikan.
3. Aplikasi SBDOJ sudah memenuhi kebutuhan fungsional .
4. *Grader* telah memenuhi kebutuhan proses *judgement*.
5. Berdasarkan *user acceptance test*, asisten membutuhkan SBDOJ untuk praktikum SBD materi DML *query select*.

6.2 Saran

Saran untuk pengembangan dan perbaikan sistem di masa yang akan datang, di antaranya adalah sebagai berikut:

1. Membuat waktu maksimal eksekusi menjadi dinamis berdasarkan soal atau jumlah *row* hasil *query* yang dibutuhkan.
2. *Grader* hendaknya dibuat mampu menangani proses *judgement* dengan menggunakan *thread* sehingga proses *judgement* menjadi lebih cepat.
3. Terdapat bobot nilai pada masing-masing soal.

DAFTAR PUSTAKA

- [1] Miguel A. Revilla, Shahriar Manzoor, Rujia Liu, “Competitive Learning in Informatics: The UVa Online Judge Experience,” *Olympiads in Informatics*, vol. 2, pp. 131-148, 2008.
- [2] Raghu Ramakrishnan, Johannes Gehrke, Database Management Systems Second Edition, 1999.
- [3] A. Rohman, “Mengenai Framework “Laravel”,” *Ilumuti*, 2014.
- [4] Y. Purdianto, “Pengertian PHP,” [Online]. Available: <http://yadicucuklauk.blogspot.co.id/2012/01/pengenalan-php.html>. [Diakses 15 Mei 2016].
- [5] D. N. Prabawa, “Pengertian Oracle,” [Online]. Available: <http://dibyanp.blogspot.co.id/2014/11/pengertian-oracle.html>. [Diakses 15 Mei 2016].

[Halaman ini sengaja dikosongkan]

LAMPIRAN A. KODE SUMBER

```
1. $data = Input::all();
2. User::create(array(
3.     'nama' => $data['nama'],
4.     'username' => $data['username'],
5.     'password' => bcrypt($data['password']),
6.     'kelas' => $data['kelas'],
7.     'role_id' => $data['role_id']
8. ));
```

Kode Sumber A.1 Fungsi Menambah Data Praktikan

```
1. $data = Input::all();
2. User::where('id', $id)->update(array(
3.     'nama' => $data['nama'],
4.     'username' => $data['username'],
5.     'role_id' => $data['role_id']
6. ));
```

Kode Sumber A.2 Fungsi Mengubah Data Praktikan

```
1. User::where('id', $id)->delete();
2. Submission::where('users_id', $id)->delete();
```

Kode Sumber A.3 Fungsi Menghapus Data Praktikan

```
1. ini_set("upload_max_filesize", "300M");
2. ini_set("post_max_size", "300M");
3. set_time_limit(10800);
4.
5. $input = Input::all();
6.
7. if ($_FILES["praktikan"]["error"] > 0)
8. {
9.     echo "Error: " . $_FILES["praktikan"]["error"] . "<br />";
10. }
11. else
12. {
```

```

13.     $praktikan = array_get($input, 'praktikan');
14.     $fileName_praktikan = $praktikan-
>getClientOriginalName();
15.     $upload_success = $praktikan-
>move("upload", $fileName_praktikan);
16.     $data = Excel::load('upload/'.$fileName_praktikan, functi
on($reader){
17.         $temp = $reader->toObject();
18.         $temp->each(function($data) {
19.
20.             $cek = User::where('username', $data->nrp)-
>count();
21.             if ($cek > 0)
22.             {
23.                 User::where('username', $data->nrp)-
>update(array(
24.                     'nama'      => $data->nama,
25.                     'kelas'     => $data->kelas,
26.                     'password' => bcrypt($data->nrp)
27.                 ));
28.             }
29.             else
30.             {
31.                 User::create(array(
32.                     'username' => $data->nrp,
33.                     'nama'     => $data->nama,
34.                     'kelas'    => $data->kelas,
35.                     'password' => bcrypt($data->nrp),
36.                     'role_id'  => 3
37.                 ));
38.             }
39.         });
40.         DB::commit();
41.     });
42.     Session::flash('status', 'success');
43.     return redirect('accounts');
44. }
45. return redirect('login');

```

Kode Sumber A.4 Fungsi Menambah Data Praktikan dengan File Excel


```

1. $data = Input::all();
2. $fileName = '';
3. if(Input::hasFile('pdm')){
4.     $file=Input::file('pdm');
5.     $extension = $file->getClientOriginalExtension();
6.     if($extension != 'jpeg' && $extension != 'JPEG'&& $extension != 'jpg' && $extension != 'JPG' && $extension != 'png' && $extension != 'PNG') {
7.         return redirect('databases/add')->withInput()-
>withErrors("Gambar PDM Harus berupa file .JPG, .PNG atau .JPEG");
8.     }
9.     $latest = ListDB::orderBy('id','desc')->first();
10.    $latest = $latest->id + 1;
11.    $name= $latest.'_pdm';
12.    $fileName = $name . '.' . $extension;
13.    $file->move(public_path().'/pdm_db',$fileName);
14. }
15. else {
16.     return redirect('databases/add')->withInput()-
>withErrors("Gambar PDM Harus diisi");
17. }
18. $db_id = ListDB::insertGetId(array(
19.     'dbversion_id'          => $data['dbversion'],
20.     'pdm'                   => $fileName,
21.     'status'                => 0,
22.     'gradertutorial_status' => 0
23. ));
24. $dbversion = $data['dbversion'];
25. array_forget($data,'dbversion');
26. array_forget($data,'_token');
27. array_forget($data,'pdm');
28. foreach ($data as $key => $value) {
29.     ListDBParameter::create(array(
30.         'listdb_id'          => $db_id,
31.         'dbversion_id'       => $dbversion,
32.         'dbversionparameter_id' => $key,
33.         'content'            => $value
34.     ));
35. }

```

Kode Sumber A.5 Fungsi Menambah Data Database

```

1. $data = Input::all();
2. $fileName = '';
3. if(Input::hasFile('pdm')){
4.     $file=Input::file('pdm');
5.     $extension = $file->getClientOriginalExtension();
6.     if($extension != 'jpeg' && $extension != 'JPEG' && $extension != 'jpg' && $extension != 'JPG' && $extension != 'png' && $extension != 'PNG')
7.     {
8.         return redirect('databases/edit/'.$id)->withInput()->withErrors("Gambar PDM Harus berupa file .JPG, .PNG atau .JPG");
9.     }
10.    $name= $id.'_pdm';
11.    $fileName = $name . '.' . $extension;
12.    $file->move(public_path().'/pdm_db',$fileName);
13.    ListDB::where('id',$id)->update(array(
14.        'pdm' => $fileName,
15.    ));
16. }
17. $dbversion = $data['dbversion'];
18. array_forget($data,'dbversion');
19. array_forget($data,'dbversion_name');
20. array_forget($data,'token');
21. array_forget($data,'pdm');
22. foreach ($data as $key => $value) {
23.     ListDBParameter::where('listdb_id',$id)->where('dbversion_id',$dbversion)->where('dbversionparameter_id',$key)->update(array(
24.         'content' => $value
25.     ));
26. }

```

Kode Sumber A.6 Fungsi Mengubah Data Database

```

1. $events = array();
2. $listEvent = Event::where('listdb_id', $id)->get();
3. foreach ($listEvent as $key) {
4.     array_push($events, $key->id);
5. }
6. ListDB::where('id',$id)->delete();
7. ListDBParameter::where('listdb_id',$id)->delete();

```

```

8. Event::where('listdb_id', $id)->delete();
9. Question::whereIn('event_id', $events)->delete();

```

Kode Sumber A.7 Fungsi Menghapus Data *Database*

```

1. $data = Input::all();
2. $id = DBversion::insertGetId(array(
3.     'nama'      => $data['nama'],
4.     'import'    => $data['import'],
5. ));
6. for ($i=1; $i <= $data['sum']; $i++) {
7.     if($data['param'][$i] != "" )
8.     {
9.         DBversionParameter::create(array(
10.             'dbversion_id' => $id,
11.             'parameter'    => $data['param'][$i]
12.         ));
13.     }
14. }

```

Kode Sumber A.8 Fungsi Menambah Jenis *Database*

```

1. $data = Input::all();
2. $id = DBversion::where('id', $id)->update(array(
3.     'nama'      => $data['nama'],
4.     'import'    => $data['import'],
5. ));
6. array_forget($data, 'nama');
7. array_forget($data, 'import');
8. array_forget($data, '_token');
9. foreach ($data as $key => $value) {
10.     if($value != "" ) {
11.         DBversionParameter::where('id', $key)-
12.         >update(array(
13.             'parameter'    => $value
14.         ));
15.     }

```

Kode Sumber A.9 Fungsi Mengubah Jenis *Database*

```

1. $data = Input::all();
2. $temp_tgl_mulai = DateTime::createFromFormat('d/m/Y', $data['
   tgl_akhir'])->format('Y-m-d');
3. $temp_tgl_akhir = DateTime::createFromFormat('d/m/Y', $data['
   tgl_mulai'])->format('Y-m-d');
4. $data['waktu_mulai'] = $temp_tgl_mulai." ".$data['wkt_mulai']
   ;
5. $data['waktu_akhir'] = $temp_tgl_akhir." ".$data['wkt_akhir']
   ;
6. Event::create(array(
7.     'judul'      => $data['judul'],
8.     'konten'      => $data['konten'],
9.     'waktu_mulai' => $data['waktu_mulai'],
10.    'waktu_akhir' => $data['waktu_akhir'],
11.    'kelas'       => $data['kelas'],
12.    'listdb_id'   => $data['db'],
13. ));

```

Kode Sumber A.10 Fungsi Menambah Praktikum

```

1. $data = Input::all();
2. $temp_tgl_mulai = DateTime::createFromFormat('d/m/Y', $data['
   tgl_mulai'])->format('Y-m-d');
3. $temp_tgl_akhir = DateTime::createFromFormat('d/m/Y', $data['
   tgl_akhir'])->format('Y-m-d');
4. $data['waktu_mulai'] = $temp_tgl_mulai." ".$data['wkt_mulai']
   ;
5. $data['waktu_akhir'] = $temp_tgl_akhir." ".$data['wkt_akhir']
   ;
6. Event::where('id', $id)->update(array(
7.     'judul'      => $data['judul'],
8.     'konten'      => $data['konten'],
9.     'waktu_mulai' => $data['waktu_mulai'],
10.    'waktu_akhir' => $data['waktu_akhir'],
11.    'kelas'       => $data['kelas'],
12.    'listdb_id'   => $data['db']
13. ));

```

Kode Sumber A.11 Fungsi Mengubah Praktikum

```

1. Event::where('id', $id)->delete();
2. Question::where('event_id', $id)->delete();

```

Kode Sumber A.12 Fungsi Menghapus Praktikum

```

1. $data = Input::all();
2. $temp_tgl_mulai = DateTime::createFromFormat('d/m/Y', $data['
   tgl_akhir'])->format('Y-m-d');
3. $temp_tgl_akhir = DateTime::createFromFormat('d/m/Y', $data['
   tgl_mulai'])->format('Y-m-d');
4. $data['waktu_mulai'] = $temp_tgl_mulai." ".$data['wkt_mulai'];
5. $data['waktu_akhir'] = $temp_tgl_akhir." ".$data['wkt_akhir'];
6. $ev_id = EventTutorial::create(array(
7.     'judul'         => $data['judul'],
8.     'konten'         => $data['konten'],
9.     'waktu_mulai'    => $data['waktu_mulai'],
10.    'waktu_akhir'    => $data['waktu_akhir'],
11.    'kelas'          => $data['kelas'],
12.    'listdb_id'       => $data['db'],
13. ));

```

Kode Sumber A.13 Fungsi Menambah Tutorial

```

1. $data = Input::all();
2. $temp_tgl_mulai = DateTime::createFromFormat('d/m/Y', $data['
   tgl_mulai'])->format('Y-m-d');
3. $temp_tgl_akhir = DateTime::createFromFormat('d/m/Y', $data['
   tgl_akhir'])->format('Y-m-d');
4. $data['waktu_mulai'] = $temp_tgl_mulai." ".$data['wkt_mulai'];
5. $data['waktu_akhir'] = $temp_tgl_akhir." ".$data['wkt_akhir'];
6. Event::where('id', $id)->update(array(
7.     'judul'         => $data['judul'],
8.     'konten'         => $data['konten'],
9.     'waktu_mulai'    => $data['waktu_mulai'],
10.    'waktu_akhir'    => $data['waktu_akhir'],
11.    'kelas'          => $data['kelas'],
12.    'listdb_id'       => $data['db']
13. ));

```

Kode Sumber A.14 Fungsi Mengubah Tutorial

```
1. EventTutorial::where('id', $id)->delete();
```

Kode Sumber A.15 Fungsi Menghapus Tutorial

```
1. $data = Input::all();
2. $check = Category::where('name', $data['name'])->count();
3. if($check > 0)
4. {
5.     Session::flash('status', 'exist');
6. }
7. else
8. {
9.     Category::create(array(
10.         'name'           => $data['name'],
11.         'detail'          => $data['detail']
12.     ));
13. }
```

Kode Sumber A.16 Fungsi Menambah Kategori Tutorial

```
1. $data = Input::all();
2. $check = Category::where('name', $data['name'])->count();
3. if($check > 0)
4. {
5.     Session::flash('status', 'exist');
6. }
7. else
8. {
9.     Category::where('id', $id)->update(array(
10.         'name'           => $data['name'],
11.         'detail'          => $data['detail']
12.     ));
13. }
```

Kode Sumber A.17 Fungsi Mengubah Kategori Tutorial

```
1. Category::where('id', $id)->delete();
```

Kode Sumber A.18 Fungsi Menghapus Kategori Tutorial

```

1. $data = Input::all();
2. Question::create(array(
3.     'event_id' => $id,
4.     'judul'    => $data['judul'],
5.     'konten'   => $data['konten'],
6.     'jawaban'  => $data['jawaban']
7. ));

```

Kode Sumber A.19 Fungsi Menambah Soal Praktikum

```

1. $data = Input::all();
2. Question::where('id', $id2)->update(array(
3.     'judul' => $data['judul'],
4.     'konten' => $data['konten'],
5.     'jawaban' => $data['jawaban']
6. ));

```

Kode Sumber A.20 Fungsi Mengubah Soal Praktikum

```

1. Question::where('id', $id2)->delete();
2. Submission::where('question_id', $id2)->delete();

```

Kode Sumber A.21 Fungsi Menghapus Soal Praktikum

```

1. $data = Input::all();
2. Tutorial::create(array(
3.     'eventttutorial_id' => $id,
4.     'category_id'       => $data['category'],
5.     'judul'             => $data['judul'],
6.     'konten'            => $data['konten'],
7.     'jawaban'           => $data['jawaban']
8. ));

```

Kode Sumber A.22 Fungsi Menambah Soal Tutorial

```

1. $data = Input::all();
2. Tutorial::where('id',$id2)->update(array(
3.     'category_id' => $data['category'],
4.     'judul'       => $data['judul'],
5.     'konten'      => $data['konten'],
6.     'jawaban'     => $data['jawaban']
7. ));

```

Kode Sumber A.23 Fungsi Mengubah Soal Tutorial

```

1. Tutorial::where('id', $id2)->delete();
2. SubmissionTutorial::where('tutorial_id',$id2)->delete();

```

Kode Sumber A.24 Fungsi Menghapus Soal Tutorial

```

1. #!/usr/bin/python
2. import MySQLdb
3. import cx_Oracle
4. import time
5. import sys
6. try:
7.     db_kunci = cx_Oracle.connect(user='sbdobj', password='wdvr
dx24680', dsn='localhost:1521/xe')
8.     cursor_kunci = db_kunci.cursor()
9.     while True:
10.         db= MySQLdb.connect('localhost', 'root', '', 'sbd')
11.         cursor = db.cursor()
12.
13.         try:
14.             sql = '''select id, question_id, users_id, jawa
ban,status from submission where status = 0 having id = min(i
d)'''
15.             cursor.execute(sql)
16.             unchecked = cursor.fetchone()
17.             sub_id = ''
18.             ques_id = ''
19.             user_id = ''
20.             ans = ''
21.             stat = ''
22.             tanda = 0
23.
24.             while unchecked is not None:
25.                 sub_id = unchecked[0]
26.                 ques_id = unchecked[1]

```



```

27.             user_id = unchecked[2]
28.             ans = unchecked[3]
29.             stat = unchecked[4]
30.             unchecked = cursor.fetchone()
31.
32.             if (sub_id!='') :
33.                 tanda = 1
34.
35.             if (tanda==1):
36.                 try:
37.                     cursor_kunci.execute(ans)
38.                 except:
39.                     pass
40.                 results = cursor_kunci.fetchall()
41.                 num_fields = len(cursor_kunci.description)
42.                 hasil=[[0 for x in range(num_fields)] for x i
n range(10000)]
43.                 rows=0
44.                 for res in results:
45.                     for column in range(num_fields):
46.                         hasil[rows][column] = res[column]
47.                         rows+=1
48.                         kunci = '''select jawaban from question
where id='''+ str(ques_id)
49.                         cursor.execute(kunci)
50.                         temp = cursor.fetchone()
51.                         while temp is not None:
52.                             temp_kunci = temp[0]
53.                             temp_kunci = temp_kunci.replace(";",
""")
54.                             temp = cursor.fetchone()
55.                             cursor_kunci.execute(temp_kunci)
56.                             res_kunci = cursor_kunci.fetchall()
57.                             num_fields_1 = len(cursor_kunci.descripti
on)
58.                             arr_kunci=[[0 for x in range(num_fields_1
)] for x in range(10000)]
59.                             row_kunci=0
60.                             for res_key in res_kunci:
61.                                 for column_kunci in range(num_fields_
1):
62.                                     arr_kunci[row_kunci][column_kunci
] = res_key[column_kunci]
63.                                     row_kunci+=1
64.                                     flag=0
65.                                     if (num_fields != num_fields_1):
66.                                         flag=1
67.

```

```

68.             if (rows != row_kunci):
69.                 flag=1
70.             if (flag==0):
71.                 for row_compare in range(row_kunci):
72.                     for column_compare in range(num_f
73.                         ields):
74.                             if (hasil[row_compare][column
75.                                 _compare]!=arr_kunci[row_compare][column_compare]):
76.                                     flag=1
77.                             if (flag==1):
78.                                 update1 = '''update submission set nila
79.                                     i = 0, status = 1 where id = ''' +str(sub_id)
80.                                     cursor.execute(update1)
81.                                     db.commit()
82.                                     print 'query ' +str(sub_id)+' failed'
83.                             elif (flag==0):
84.                                 update2 = '''update submission set nila
85.                                     i = 100, status = 1 where id = ''' +str(sub_id)
86.                                     cursor.execute(update2)
87.                                     db.commit()
88.                                     print 'query ' +str(sub_id)+' success'
89.         except :
90.             print 'Exception ' +str(sub_id)
91.             db.rollback()
92.             update1 = '''update submission set nilai = 0, s
93.                 tatus = 1 where id = ''' +str(sub_id)
94.                 cursor.execute(update1)
95.                 db.commit()
96.                 time.sleep(1)
97.                 db.close()
98.                 db_kunci.close()
99.     except KeyboardInterrupt:
100.         sys.exit(0)
101. except :
102.     print 'berhenti'

```

Kode Sumber A.25 Grader versi I

```

149.     #!/usr/bin/python\
150.     import sys
151.     import os
152.     from multiprocessing import Process
153.     import time
154.     import threading

```



```

190.             hasil[rows][column] = res[co
            lumn]
191.             rows+=1
192.
193.
194.             kunci = '''select jawaban from quest
            ion where id=''+ str(ques_id)
195.             cursor.execute(kunci)
196.             temp = cursor.fetchone()
197.             while temp is not None:
198.                 temp_kunci = temp[0]
199.                 temp_kunci = temp_kunci.replace(
                ';;', '')
200.                 temp = cursor.fetchone()
201.                 cursor_kunci.execute(temp_kunci)
202.                 res_kunci = cursor_kunci.fetchall()
203.                 num_fields_1 = len(cursor_kunci.desc
                ription)
204.                 arr_kunci=[[0 for x in range(num_fie
                lds_1)] for x in range(10000)]
205.                 row_kunci=0
206.                 for res_key in res_kunci:
207.                     for column_kunci in range(num_fi
                elds_1):
208.                         arr_kunci[row_kunci][column_
                kunci] = res_key[column_kunci]
209.                         row_kunci+=1
210.
211.
212.                 flag=0
213.                 if (num_fields != num_fields_1):
214.                     flag=1
215.                 if (rows != row_kunci):
216.                     flag=1
217.                 if (flag==0):
218.                     for row_compare in range(row_kun
                ci):
219.                         for column_compare in range(
                num_fields):
220.                             if (hasil[row_compare][c
                olumn_compare]!=arr_kunci[row_compare][column_compa
                re]):

```



```

255.         updateTLE = '''update submission set nil
ai = 101, status = 1 where id = ''' + str(sub_id)
256.         cursor.execute(updateTLE)
257.         db.commit()
258.         p.terminate()
259.
260.         if __name__ == '__main__':
261.             try:
262.                 while True:
263.                     db= MySQLdb.connect('localhost',
'root', '', 'sbd')
264.                     cursor = db.cursor()
265.                     #try:
266.                     sql = '''select s.id, s.question
_id, s.users_id, s.jawaban, s.status from submissio
n s,event e, listdb ldb, question q where s.questio
n_id = q.id and q.event_id = e.id and e.listdb_id =
ldb.id and s.status = 0 and ldb.dbversion_id = '''
+sys.argv[2]+''' having s.id = min(s.id)'''
267.                     cursor.execute(sql)
268.                     unchecked = cursor.fetchone()
269.                     sub_id = ''
270.                     ques_id = ''
271.                     user_id = ''
272.                     ans = ''
273.                     stat = ''
274.                     tanda = 0
275.                     while unchecked is not None:
276.                         sub_id = unchecked[0]
277.                         ques_id = unchecked[1]
278.                         user_id = unchecked[2]
279.                         ans = unchecked[3]
280.                         stat = unchecked[4]
281.                         unchecked = cursor.fetchone(
)
282.                     if (sub_id!='') :
283.                         tanda = 1
284.                     if (tanda==1):
285.                         t = threading.Thread(target=
stopwatch, args=(20,check,(str(sub_id),)))
286.                         t.start()
287.                         time.sleep(1)
288.                         t.join()

```

```
289.             print "next"
290.             db.close()
291.             db_kunci.close()
292.
293.         except KeyboardInterrupt:
294.             sys.exit(0)
295.     except :
296.         print 'berhenti'
```

Kode Sumber A.26 Grader versi II

LAMPIRAN B. KUESIONER UAT

User Acceptance Test Aplikasi Sistem Basis Data Online Judge

Sehubungan dengan Tugas Akhir saya, Demsy Iman Mustasyar, saya meminta sedikit waktu Anda sebagai asisten SBD untuk mengisi kuisioner User Acceptance Test aplikasi SBDOJ. Berikut keterangan yang dapat digunakan sebagai panduan mengisi kuisioner ini beserta kuisioner dan biodata Anda yang saya harap Anda tidak keberatan untuk mengisinya.

Tabel Bobot Nilai Jawaban

Jawaban	Bobot
STS = Sangat Tidak Setuju	1
TS = Tidak Setuju	2
N = Netral	3
S = Setuju	4
SS = Sangat Setuju	5

Nama : Bima Nisrina Majid

NRP : 5112100019

No	Pertanyaan	STS	TS	N	S	SS
1	SBDOJ dapat dioperasikan dengan mudah oleh asisten				v	
2	SBDOJ sudah memenuhi kebutuhan praktikum SBD materi DML query select				v	
3	SBDOJ memudahkan asisten dalam praktikum SBD materi DML query select					v
4	SBDOJ dapat menghemat waktu praktikum SBD					v
5	Proses penilaian SBDOJ sudah benar				v	
6	Asisten membutuhkan SBDOJ untuk praktikum SBD materi DML query select					v
7	SBDOJ hendaknya digunakan seterusnya pada saat praktikum SBD materi DML query select					v

Atas waktu dan kesediaan Anda dalam membantu Tugas Akhir ini, saya ucapkan terimakasih.

Lampiran B. 1 Kuesioner *User Acceptance Test 1*

User Acceptance Test Aplikasi Sistem Basis Data Online Judge

Sehubungan dengan Tugas Akhir saya, Demsy Iman Mustasyar, saya meminta sedikit waktu Anda sebagai asisten SBD untuk mengisi kuisoner User Acceptance Test aplikasi SBDOJ. Berikut keterangan yang dapat digunakan sebagai panduan mengisi kuisoner ini beserta kuisoner dan biodata Anda yang saya harap Anda tidak keberatan untuk mengisinya.

Tabel Bobot Nilai Jawaban

Jawaban	Bobot
STS = Sangat Tidak Setuju	1
TS = Tidak Setuju	2
N = Netral	3
S = Setuju	4
SS = Sangat Setuju	5

Nama : Hifnie Bilfash

NRP : 5113100091

No	Pertanyaan	STS	TS	N	S	SS
1	SBDOJ dapat dioperasikan dengan mudah oleh asisten				v	
2	SBDOJ sudah memenuhi kebutuhan praktikum SBD materi DML query select				v	
3	SBDOJ memudahkan asisten dalam praktikum SBD materi DML query select				v	
4	SBDOJ dapat menghemat waktu praktikum SBD				v	
5	Proses penilaian SBDOJ sudah benar				v	
6	Asisten membutuhkan SBDOJ untuk praktikum SBD materi DML query select				v	
7	SBDOJ hendaknya digunakan seterusnya pada saat praktikum SBD materi DML query select					v

Atas waktu dan kesediaan Anda dalam membantu Tugas Akhir ini, saya ucapkan terimakasih.

Lampiran B. 2 Kuesioner User Acceptance Test 2

User Acceptance Test Aplikasi Sistem Basis Data Online Judge

Sehubungan dengan Tugas Akhir saya, Demsy Iman Mustasyar, saya meminta sedikit waktu Anda sebagai asisten SBD untuk mengisi kuisoner User Acceptance Test aplikasi SBDOJ. Berikut keterangan yang dapat digunakan sebagai panduan mengisi kuisoner ini beserta kuisoner dan biodata Anda yang saya harap Anda tidak keberatan untuk mengisinya.

Tabel Bobot Nilai Jawaban

Jawaban	Bobot
STS = Sangat Tidak Setuju	1
TS = Tidak Setuju	2
N = Netral	3
S = Setuju	4
SS = Sangat Setuju	5

Nama : Dwi Al Aji Suseno

NRP : 5112100185

No	Pertanyaan	STS	TS	N	S	SS
1	SBDOJ dapat dioperasikan dengan mudah oleh asisten					v
2	SBDOJ sudah memenuhi kebutuhan praktikum SBD materi DML query select					v
3	SBDOJ memudahkan asisten dalam praktikum SBD materi DML query select					v
4	SBDOJ dapat menghemat waktu praktikum SBD					v
5	Proses penilaian SBDOJ sudah benar					v
6	Asisten membutuhkan SBDOJ untuk praktikum SBD materi DML query select					v
7	SBDOJ hendaknya digunakan seterusnya pada saat praktikum SBD materi DML query select					v

Atas waktu dan kesediaan Anda dalam membantu Tugas Akhir ini, saya ucapkan terimakasih.

Lampiran B. 3 Kuesioner *User Acceptance Test 3*

User Acceptance Test Aplikasi Sistem Basis Data Online Judge

Sehubungan dengan Tugas Akhir saya, Demsy Iman Mustasyar, saya meminta sedikit waktu Anda sebagai asisten SBD untuk mengisi kuisoner User Acceptance Test aplikasi SBDOJ. Berikut keterangan yang dapat digunakan sebagai panduan mengisi kuisoner ini beserta kuisoner dan biodata Anda yang saya harap Anda tidak keberatan untuk mengisinya.

Tabel Bobot Nilai Jawaban

Jawaban	Bobot
STS = Sangat Tidak Setuju	1
TS = Tidak Setuju	2
N = Netral	3
S = Setuju	4
SS = Sangat Setuju	5

Nama : Renault Silva Natalensis S

NRP : 5113100142

No	Pertanyaan	STS	TS	N	S	SS
1	SBDOJ dapat dioperasikan dengan mudah oleh asisten				v	
2	SBDOJ sudah memenuhi kebutuhan praktikum SBD materi DML query select				v	
3	SBDOJ memudahkan asisten dalam praktikum SBD materi DML query select				v	
4	SBDOJ dapat menghemat waktu praktikum SBD				v	
5	Proses penilaian SBDOJ sudah benar					v
6	Asisten membutuhkan SBDOJ untuk praktikum SBD materi DML query select					v
7	SBDOJ hendaknya digunakan seterusnya pada saat praktikum SBD materi DML query select					v

Atas waktu dan kesediaan Anda dalam membantu Tugas Akhir ini, saya ucapkan terimakasih.

Lampiran B. 4 Kuesioner *User Acceptance Test* 4

User Acceptance Test Aplikasi Sistem Basis Data Online Judge

Sehubungan dengan Tugas Akhir saya, Demsy Iman Mustasyar, saya meminta sedikit waktu Anda sebagai asisten SBD untuk mengisi kuisoner User Acceptance Test aplikasi SBDOJ. Berikut keterangan yang dapat digunakan sebagai panduan mengisi kuisoner ini beserta kuisoner dan biodata Anda yang saya harap Anda tidak keberatan untuk mengisinya.

Tabel Bobot Nilai Jawaban

Jawaban	Bobot
STS = Sangat Tidak Setuju	1
TS = Tidak Setuju	2
N = Netral	3
S = Setuju	4
SS = Sangat Setuju	5

Nama : Rina Wijaya Kusuma Wardhani

NRP : 5114100021

No	Pertanyaan	STS	TS	N	S	SS
1	SBDOJ dapat dioperasikan dengan mudah oleh asisten				v	
2	SBDOJ sudah memenuhi kebutuhan praktikum SBD materi DML query select				v	
3	SBDOJ memudahkan asisten dalam praktikum SBD materi DML query select				v	
4	SBDOJ dapat menghemat waktu praktikum SBD					v
5	Proses penilaian SBDOJ sudah benar			v		
6	Asisten membutuhkan SBDOJ untuk praktikum SBD materi DML query select				v	
7	SBDOJ hendaknya digunakan seterusnya pada saat praktikum SBD materi DML query select					v

Atas waktu dan kesediaan Anda dalam membantu Tugas Akhir ini, saya ucapkan terimakasih.

Lampiran B. 5 Kuesioner User Acceptance Test 5

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS



Demsy Iman Mustasyar, lahir di Lumajang, pada tanggal 20 Februari 1995. Penulis menempuh pendidikan formal mulai dari TK Bhayangkara Lumajang (2000-2001), SDN Ditotrunan 1 Lumajang (2001-2007) SMP Negeri 1 Lumajang (2007-2010), SMA Negeri 2 Lumajang (2010-2013) dan S1 Teknik Informatika Fakultas Teknologi Informasi di Institut Teknologi Sepuluh Nopember Surabaya (2013-2017).

Selama masa kuliah, penulis aktif dalam organisasi Himpunan Mahasiswa Teknik Computer (HMTC). Diantaranya adalah menjadi staf Departemen Pengembangan Sumber Daya Mahasiswa HMTC ITS 2014-2015. Penulis juga aktif dalam kepanitiaan SCHEMATICS ITS 2014 dengan menjadi Wakil Ketua 2.

Selama kuliah di Teknik Informatika ITS, penulis mengambil bidang minat Algoritma dan Pemrograman (Alpro) dan menjadi administrator di Laboratorium Algoritma dan Pemrograman. Penulis dapat dihubungi melalui alamat surel **demsyiman@gmail.com**.